TAMPERE UNIVERSITY OF TECHNOLOGY

Zhao Shuyang

A personalized hybrid music recommender based on empirical estimation of user-timbre preference

Master of Science Thesis

Examiner: Adj.Prof. Tuomas Virtanen
Examiner and topic approved by the Faculty Council of the Faculty of Computing and Electrical Engineering on 15.04.2014

ABSTRACT

TAMPERE UNIVERSITY OF TECHNOLOGY
Master's Degree Programme in Information Technology
Signal Processing Laboratory
AUTHOR : Zhao Shuyang
Master of Science Thesis, 50 pages, 0 Appendix pages
Examiner: Adj.Prof. Tuomas Virtanen
Keywords: Personalized Music Recommendation, Ranking Prediction, Timbre Feature, Bayesian Estimation

Automatic recommendation system as a subject of machine learning has been undergoing a rapid development in the recent decade along with the trend of big data. Particularly, music recommendation is a highlighted topic because of its commercial value coming from the large music industry.

Popular online music recommendation services, including Spotify, Pandora and Last.FM use similarity-based approaches to generate recommendations. In this thesis work, I propose a personalized music recommendation approach that is based on probability estimation without any similarity calculation involved. In my system, each user gets a score for every piece of music. The score is obtained by combining two estimated probabilities of an acceptance. One estimated probability is based on the user's preferences on timbres. Another estimated probability is the empirical acceptance rate of a music piece. The weighted arithmetic mean is evaluated to be the best performing combination function.

An online demonstration of my system is available at www.shuyang.eu/plg/. Demonstrating recommendation results show that the system works effectively. Through the algorithm analysis on my system, we can see that my system has good reactivity and scalability without suffering cold start problem. The accuracy of my recommendation approach is evaluated with Million Song Dataset. My system achieves a pairwise ranking accuracy of 0.592, which outperforms random ranking (0.5) and ranking by popularity (0.557). Unfortunately, I have not found any other music recommendation method evaluated with ranking accuracy yet. As a comparison, Page Rank algorithm (for web page ranking) has a pairwise ranking accuracy of 0.567 [38].

PREFACE

This work has been conducted at the Department of Signal Processing of Tampere University of Technology.

The starting point of my study on music recommendation is an innovative project sponsored by Nokia. In this project, I implemented a collaborative filtering music recommender using random indexing instead of matrix factorization to make a trade-off between computation complexity and accuracy. Along with the progress of this project, I had been taking the course Speech Recognition lectured by Tuomas Virtanen. This course aroused my interest in audio processing and with my personal aversion against prevailing similarity-based music recommendation services, I developed the basic idea of the music recommender proposed in this thesis work.

This thesis work is my first publication written in English and also my first one in Latin alphabet. As is often the case, one meets a lot of difficulties for the first time and grows stronger after overcoming them. I generally formed an image of what scientific research is during this thesis work. I draw myself a brief conclusion that scientific research is not a sport since research is not competition of how fast you understand and implement an algorithm. More important thing in scientific research is to follow a normalized manner or called scientific method to draw conclusion and present .

Finally, I should thank Tuomas Virtanen as my supervisor for reading and correcting my thesis. Furthermore, hopefully there is a spot of light in my thesis that interests you.

Zhao Shuyang

Tampere, 02/02/2014

1. INTRODUCTION

Music recommendation is an interdisciplinary subject, which involves machine learning and music information retrieval. Following paragraphs talk briefly about the role and history of automatic recommendation system and the rise of music recommendation as a problem.

Internet is providing a huge amount of information, as is shown by Google statistics that trillion(10^{10}) pages are being indexed in 2011. Search engines enable users to make specific queries for information. Parallel to search engines, recommendation systems filter for information that may interest users without specific query. The core functionality of automatic recommendation systems is be discussed in Section 2.1.2. First recommendation system is an online news recommender called Tapestry by Goldberg, which emerged in 1992. Nowadays, many large scale e-commercial sites, including Amazon, Netflix and TiVo run recommendation systems to mine potential purchase interest of their customers to increase their sales. Besides prompt to sales, recommendation systems also help to build customer loyalty, and to open up advertising revenues. GroupLens [21] is an early instance of collaborative filtering recommendation. Collaborative filtering still prevails today, Amazon being using as an example. Collaborative filtering is generic to all types no matter if the content is news, movies or music. Content-specific recommendation techniques are also developed to meet higher requirements of system performance, including accuracy and scalability.

With the development of digital storage technique and increasing network bandwidth in the recent decade, multimedia data started to play an important role on the internet, which used to be dominated by textual data, especially on mobile devices. The advancement of online multimedia services raised a challenge on multimedia information retrieval. ISMIR (International Seminar of Music Information Retrieval) started in 2000 and is held annually on the topic of music information retrieval. An example task of music information retrieval is as follows: a user sings a segment of a song, which is recorded, then the record is used as a request to query for the song title and the artist. Music information retrieval techniques provide various approaches that access a wide range of descriptors of music, which makes music-specific recommendation method very promising. Besides available techniques, there is also a demand for music-specific recommendation techniques since the magnitude of avail-

able music nowadays is high. The contemporary music industry worldwide is so productive that 10,000 new albums and 100,000 pieces of music are registered for copyright each year [9]. With such many music descriptors provided by music information retrieval techniques, there are large number of possible solutions to make music recommendations. To study on the possible solutions, Million Song Dataset Challenge was raised 2012 under the data science site Kaggle [1] for predicting what users will listen given their listening history. This contest is well known and their evaluation rule is widely understood. As is stated in the official publication of Million Song Dataset challenge [15], the challenge is a large scale, personalized music recommendation challenge that is to predict the songs user will listen to. Million Song Dataset is used to evaluate my system and the evaluation method and the results are presented in Chapter 5.

1.1 Objectives and Main Results

The main objective of this thesis is to propose a novel music recommendation method, which is computationally cheap. The system has good reactivity and suffers no cold start problem. The performance of my recommendation system is evaluated with ranking prediction metric based on Million Song Dataset. The ranking accuracy of my system is 0.592 that clearly outperforms random ranking (theoretically 0.5) and ranking by popularity (0.557). Most music recommendation algorithm is not evaluated with ranking accuracy so that it is difficult to make comparison. Pairwise ranking accuracy is more commonly used for web page ranking. The famous Page Rank algorithm has a ranking accuracy of 0.567 [38].

1.2 Organization of the Thesis

The thesis starts with background information about techniques related to music recommendation system in Chapter 2. The background information includes the typology of automatic recommendation systems, a review of music information retrieval techniques, introduction of the Gaussian mixture model (GMM) and several similarity-based recommendation techniques. Chapter 3 introduces my recommendation approach and its complexity is analyzed in Chapter 4. Chapter 5 evaluates the performance of my system.

[1] http://www.kaggle.com

2. BACKGROUND

This chapter starts with the taxonomy of recommendation system and popular online music recommendation services are discussed after that. Furthermore, a review on music information retrieval briefly covers a wide range of topics in this area. MFCCs (Mel-frequency cepstral coefficients) and GMMs are introduced with more details since they play important role in my recommendation system and some relative researches. Important music similarity metrics are then introduced since similarity-based recommendation is the mainstream at present.

2.1 Taxonomy of Recommendation System

A similarity between a recommendation system and a search engine is that they both help users to filter information. The main difference is that a recommendation system does not need a specific query that is a must for a search engine. Rao and Talwar [2] identified 96 recommendation systems on various subjects. To understand the similarities and differences of diverse recommendation systems, Eric Gaussier classifies recommendation systems by data sources and functionalists in his doctoral thesis [32]. My introduction of recommendation system follows this thread.

Before going into details, four important notions need to be explained. A user is a recognized individual who has an unique identification. The identification could either be a user name registered by the user or a unique value generated by the system. An item is a conceptual unit of content in a system and there is usually metadata describing items. For example, in Amazon an item is a product and in a music recommendation system an item is a piece of music. A rating is a quantized review by a user on an item. Rating is also called voting in some cases. The word access is used when a user acts on an item. Purchase history and listen history are two instances of access history.

2.1.1 Data Sources

There are basically three types of data sources for recommendation system. They are rating histories (or access histories), users' features and items' features. There are different rating scales. For example, the 1-5 scale with 5 stars representation is used by many sites such as Amazon, CNet and hotels.com. Another widely used rating scale is a binary scale with its presentation of thumb up and thumb down

meaning like and dislike. Pre-processing is usually required to manage multiple sources. For example, merging rating data from Amazon and YouTube, a possible operation could be a transformation mapping $\{3, 4, 5\}$ from 1-5 scale to $True$ and $\{1, 2\}$ to $False$ so that whole rating data is transformed to the binary scale. $True$ and $False$ stands for like and dislike, which is linked with thumb up and thumb down action in YouTube.

The access history of items is an alternative to the rating history. The use of access history is referred to as unsupervised learning in a presentation about Spotify music recommendation system [39] whereas the use of rating history is called supervised learning . In the context of music recommendation, the access history is a listen history that is easier to collect compared to the rating history since users do not always rate after listening. Million song dataset [14] is an example of listen history dataset.

Users' features include socio-demographic data like age, gender and location [19]. Users' features could also be tags like 'metal fans' or 'fancier of military affairs' and other descriptors that represents users' characteristics [2]. Item features include intrinsic characteristics (e.g. MFCCs), textual descriptions (for example 'folk music' and '90s' pop music') and any other descriptors of an item. Possible descriptive features for music will be introduced in the next section. Figure 2.1 illustrates the three types of possible data sources. Users' and items' features may be in any data type, integer, float, text, etc. Pre-processing of data such as replacing text with a number or boolean is, in most cases, trivial, so that the pre-processing procedure is not interesting although it is important and time consuming. Since pre-processing has such a nature, this work will not go into details.

The classical music recommendation typology is based on data sources utilized in the recommendation system. Collaborative filtering (CF) uses rating matrix [19, 20, 21, 22, 23] and content-based filtering (CBF) uses item features [24, 25, 26, 27]. Hybrid filtering uses both rating matrix and item features [3]. In [2], the notion of demographic filtering is proposed, which uses scocio-demographic data such as age and location.

2.1.2 Functionality of Recommendation System

Another typology is based on the core function of the recommendation systems. It would be of great importance to design a proper functionality to meet users' demand, however few studies have been made about the functionality of recommendation system. Gaussier [32] proposed four essential core functions of recommendation systems.

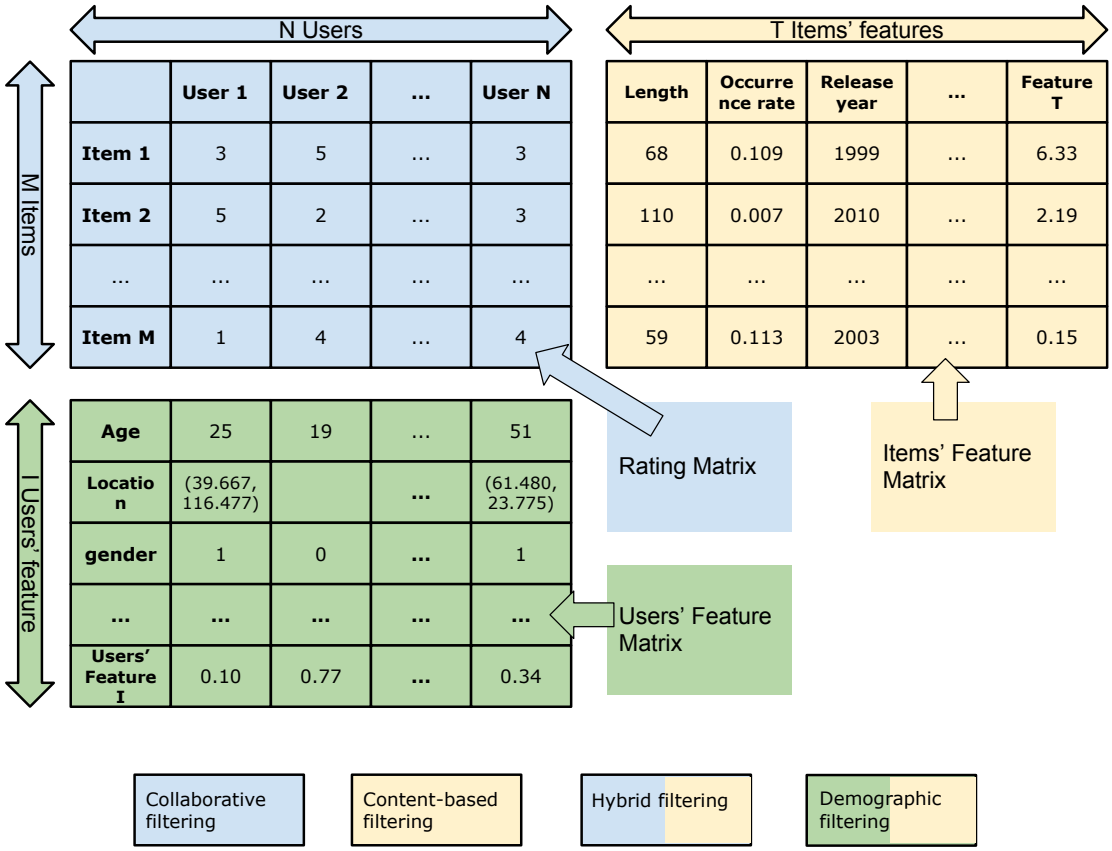

Figure 2.1: Example of data sources and corresponding recommendation system classifications.

Rating prediction A rating predicting system tries to minimize cost function of predictive error $(\hat{r} - r)$. Mean absolute error (MSE) or root mean square error (RMSE) are conventionally used as the cost function.

Rank prediction Rank predicting system tries to maximize the number of correct ranking pairs. For example, ground truth ranking order of five items is $[a, b, c, d, e]$. There are $\begin{pmatrix} 2 \\ 5 \end{pmatrix} = 10$ pairs, e.g. $[b, c]$ and $[a, e]$. If predicted ranking is $[a, c, b, e, d]$, the number of correct ranking pairs is $|\{[a, c], [a, b], [a, e], [a, d], [c, e], [c, d], [b, e], [b, d]\}| = 8$.

Contextual recommendation Contextual recommendation is also called item-to-item similarity based recommendation. One example is related videos on YouTube that is a list of top similar videos to the video that is being browsed. Another example is playlist radio on Spotify, which takes playlist as a query and randomly plays a song that is very similar to at least one r the songs in the sample playlist.

Top-N Personalized recommendation This type of recommendation systems recommend N items with a top personalized utility score calculated from an utility function $f(u, i)$. Personalized recommendation is an opposite to contextual recommendation since personalized utility function takes an user as the input whereas contextual recommender is based on item-to-item similarity.

As the earliest emerging recommendation functionality, rating prediction has many instances including Movielens[1]. Movielens requires new users to rate for at least 15 movies to generate rating predictions and the system ranks movies by rating or predicted rating value. Movielens system uses blue stars for actual ratings and red stars for predicted ratings, and ranks movies by predicted ratings.

One important application of rank prediction is the ranking of a search result. For example, Taobao is an e-commerce site that ranks products based on user profile. The details of the rank prediction algorithm of Taobao are not available but personally I suspect Taobao to run demographic filtering since geographic distance between an user and a seller has an influence on result ranking. An important publication of rank predicting recommendation is "Social ranking: Finding relevant content in Web 2.0" [35], which ranks query results by social tags. This work describes a system that users query with a set of tags and the result is ranked using a collaborative filtering method by calculating user-to-user similarity and tag-tag similarity.

Contextual recommendation has wide industrial use. Amazon started to use item-to-item contextual recommendation for products since 2003. The recommendation shows on product preview page with its name "Customers Who Bought This Item Also Bought" and Amazon also recommends by associative mining, by which the result is called "Frequently Bought Together". Last.fm runs a radio service that allows users to query by a sample song and randomly plays songs that are similar to the sample. Besides the radio service, Last.fm also recommends artists that are similar to users' favorite artists. YouTube provides both contextual recommendation and personalized recommendation. The contextual recommendation shows on the right side of a video playing page and personalized recommendation shows on a homepage with the title "Recommended for you".

The task of Million Song Challenge is not covered in the above-mentioned four functionalities. In general, the task of Million Song Challenge is access prediction. Million Song Dataset launched a challenge in the April of 2012. It gives a full listening history for one million users and half of the listening history for 110,000 users (10,000 in the validation set, 100,000 in the test set). The challenge is to predict the missing half. The task is to predict music pieces that a user will listen to in the future regardless of whether the user would like them.

[1] www.movielens.org

2.2 Popular Music Recommendation Services and My Approach

Pandora, Last.fm and Spotify are referred as popular recommendation services in some publications [14, 15]. Furthermore, they are discussed and compared by many non-academic blog authors on the internet. Pandora recognizes 65 million active users and Spotify has 24 million in 2013. In 2012, Last.fm claimed 51 million accounts. All of above mentioned 3 music intelligence services have basically two functions. One is called radio and another is similar artist recommendation. Radio means randomly playing music pieces of based on a query sample. In Pandora and Last.fm the radio playlist is generated responding to a query of an example artist so that radio playlist is random music pieces by similar artists. Their radio is referred sometimes by artist radio, whereas Spotify takes an example playlist as input to generate a radio playlist. Generally speaking, all of these three most popular music recommenders do similarity-based contextual recommendation. Arguments on similarity-based recommendation leads me to my different philosophy of doing music recommendation, which is introduced in Section 3.1.

2.3 Music Information Retrieval

Music Recommendation involves both the study of automatic recommendation and music information retrieval. Music information retrieval (MIR) is an interdisciplinary field of science that involves musicology, psychology signal processing and machine learning. Besides music recommendation, the application of MIR also includes music transcription (audio to MIDI), music generation (automatic composition and synthetization), genre classification, instrument recognition and so on.

The starting question before retrieving the information must be "What information music contains?". We all know that human decodes speech signal into semantic information and sometimes also emotional information, but how about music? The answer may varies from an expert to expert. Stephen Downie listed seven facets of music information in Annual Review of Information Science and Technology [43]. They are pitch, temporal, harmonic, timbral, editorial, textual and bibliographic facets. For a single tone, there are three features: loudness, pitch and timbre. Pitch and timbre facets deal with pitch and timbre, whereas loudness along with duration falls into temporal facet. The harmonic facet studies polyphony that is two or more pitches occurring simultaneously. The editorial facet includes fingerings, ornamentation, etc. The lyrics of a song belong to the textual facet and peripheral information out of content of music such as song title, lyric author and release date are classified as the bibliographic facet.

2.3.1 Pitch Fact

Pitch is defined as "the perceived quality of a sound that is chiefly a function of its fundamental frequency in—the number of oscillations per second" [8]. Human perception of pitch interval is approximately logarithmic with respect to fundamental frequency. An octave is the interval between one musical pitch and another with half or double its frequency. For example, the interval between 100 Hz and 200 Hz is an octave. Furthermore, the interval between 500 Hz and 1000 Hz is also an octave. A tradition in western music is to divide an octave into 12 equal semitones. As an example, a piano has 88 keys playing musical pitches with ascending order from left to right with step of 1 semitone per key. Helmholtz pitch notation uses $(C, C\#, D\#, E, F, F\#, G, G\#, A, A\#, B)$ to represent 12 semitones and this representation system is widely used by musicians across the world. MIDI tuning standard (MTS) [42] maps a fundamental frequency to a semitone level by as

$$p = 69 + 12 \times \log_2 \left(\frac{f}{440\text{Hz}} \right), \tag{2.1}$$

where p is the semitone levels and f is the frequency. Equation (2.1) sets a reference at 440 Hz frequency as middle A for 69th semitone level and calculates semitone levels of other frequencies with the reference.

A series of pitches, for instance (EEFGGFED...), forms a melody. In Harvard dictionary of music, the definition of melody is a linear succession of musical tones that the listener perceives as a single entity [8]. However, the word "melody" is sometimes ambiguous since the word may refer to both pitch series and the duration of each pitch in some cases in daily use. In this thesis, the word "melody" is used as it is defined in Harvard dictionary of music. Melody is an important identification of a music work. Two pieces of music audio are recognized as the same song if their melody is basically same and they are recognized as different versions for using different instruments or in different lyrics. Unpitched percussion instrument produces non-melodic music and such kind of percussive music is identified by its rhythm.

With such a nature, pitch (or called chromatic feature in some research) is used for music recognition research. Pitch series is commonly called progression in the field of music. From recommendation point of view, the pattern of progression, as the core part of most music pieces, necessarily associates with preference of listeners. Thus, it would be a direct solution to recommend music by its melodic pattern. However, no publication has been found that studies recommendation by melodic patterns.

2.3.2 Temporal Facet

Rhythmic information including tempo, meter, duration and accent falls under temporal facet. Tempo indicates the speed of music, which is usually measured by BPM (beats per minute). Accent, in the context of music, means emphasis placed on a music note, which can be either monophonic pitch or polyphonic harmonic. Meter as a music term means the regular repeating structure of accent. Common examples of metric structures are duple meters and triple meters. Duple meter means that accents are placed on the first beat of every two beats and triple meter means accents are placed on the first beat of every three beats. Rhythmic information is also very important component of music. The importance may vary with music culture. For example, in the traditional Chinese music culture, the rhythm is less important than in modern music since many Chinese music books record only pitch series without restricting rhythm and it is up to performers to improvise on rhythmic part. However, temporal information in modern music is clearly important and relevant to listeners' preference. A basic temporal feature to utilize on music recommendation is BPM. For example, beats per minute (BPM) can be used as in a demographic filtering with the hypothesis that teenagers like fast music whereas seniors like slow music. It is also reasonable to take meter of music into account for recommendation.

2.3.3 Timbral Facet

Timbre is defined as "an attribute of sensation in terms of which a listener can judge that two sounds having the same loudness and pitch are dissimilar" [30] by American Standards Association. It is timbre that makes piano sound different from violin and makes my voice different from others. What acoustic features contribute to timbre? The answer is not simple and exact. The definition in the beginning of this paragraph assigns all acoustic features else than pitch and loudness to be timbre and some acousticians claim timbre to be "the psychoacoustician's multidimensional waste-basket category for everything that cannot be labeled pitch or loudness" [29]. In synthetization, harmonics and envelope are two most influential concepts for timbre and they are widely used for timbre modulation in sound synthetization. Figure 2.2 and 2.3 show harmonics and envelope of piano and trumpet. From these figures, it is easy to spot some remarkable points that trumpet has rich harmonic components and piano has a longer decay time. For the analysis of timbre, mel-frequency cepstral coefficients (MFCCs) are a type of commonly used features in audio recognition and will be reviewed in 2.4.

Harmonics Harmonics are the set of frequencies produced by sinusoidal motion of an oscillating system. Both wind instruments such as trumpets and string in-

struments such as violins have large number of harmonics. An acoustic source that oscillates with multiple normal modes is called harmonic source. The frequency of the fundamental mode is called fundamental frequency. Harmonics are multiples of the fundamental frequency. Pitch value is determined by the fundamental frequency. Fundamental frequency component is usually but not always strongest in magnitude.

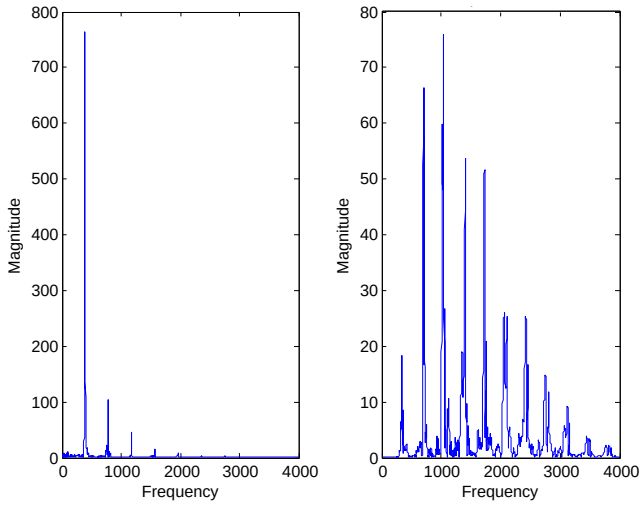

Figure 2.2: Harmonic spectrum of piano and trumpet.

Envelope Envelope, in the context of sound synthetization, is a time-amplitude function to modulate amplitude of sound change over time. An evidence that envelope contributes to timbre is that if the same single tone produced by piano and violin, are edited so that the beginning and the end of the tone are removed, they would sound similar. One method to calculate the envelope of a signal is low-pass filtering after rectifying. In Figure 2.3, audio signals produced by a piano and a trumpet are shown in blue and their envelope detected by full wave rectification and Butterworth low-pass filtering are marked red. A common model for electric music synthesizer is the ADSR model that describes the amplitude change of a single tone with four phases: attack, decay, sustain and release. Illustration of the ADSR envelope model is shown in Figure 2.3. Variations of ADSR, e.g. AHDSR (attack, hold, decay, sustain, release) and DAHDSR (delay, attack, hold, decay, sustain, release), use additional envelope modulating parameters.

In Figure 2.4, A stands for attack time, which is the time after driving force is imposed on the oscillating system and before the time that the oscillating amplitude starts to decrease. D stands for decay time, which is the time after the decrease of oscillating and before the oscillating amplitude reduce to a sustain level. S stands for the sustain level of oscillation when the driving force holds. R is the release time

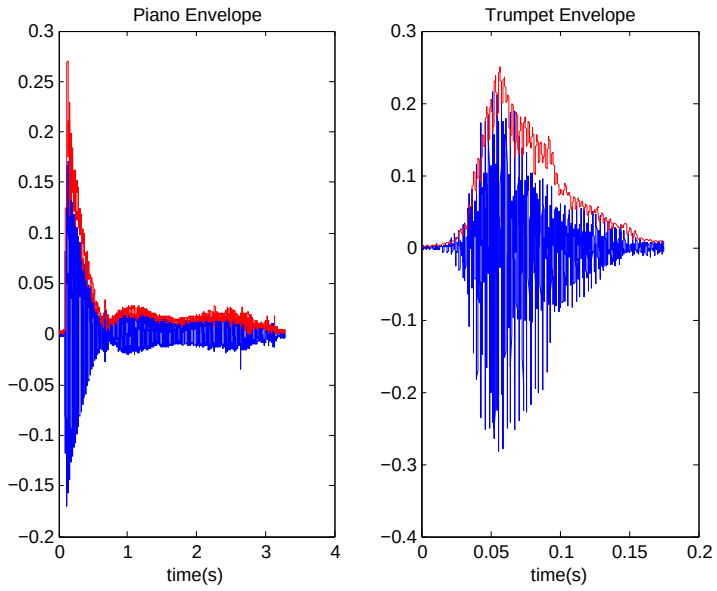

Figure 2.3: Envelope of piano and trumpet calculated with full wave rectification and Butterworth low-pass filter.

between the remove of driving force and the stop of the oscillation.

2.3.4 Other facets

Other facets in music information retrieval research have not yet been much studied for music recommendation so that this subsection introduces them in a brief way. Harmonic facet should be distinguished from concept of harmonics in timbre.

Harmony Harmony in musicology means that two or more pitches sound at the same time. The word 'chord' is more often used when three or more pitches sound simultaneously. For instance, the C major triad's is noted, C-E-G. The study of harmony has been the central part of Western classical music. The detection of a chord is simply polyphonic version of pitch detection and such technique is used in music recognition [17, 18] and music game, e.g. Wild Chord by Ovelin. Wild Chord is an Ipad game that instructs guitar practicers to play a series of chords with timeline and detect their correctness. A chord progression pattern, as well as single tone progression pattern might be potential for music recommendation. However, most modern popular music only use chords for accompaniment, which is unlikely to be an important aspect of music for user preference.

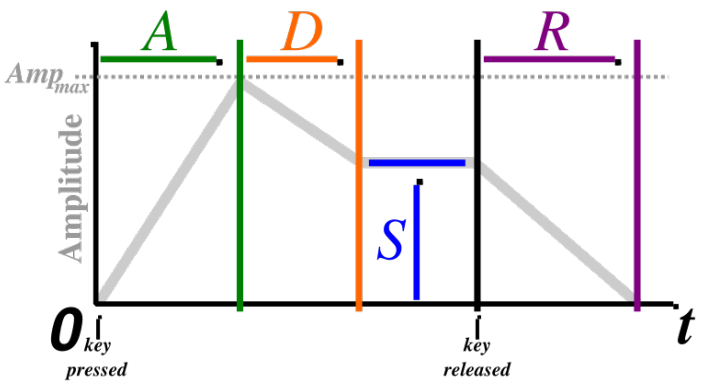

Figure 2.4: ADSR parameters [40].

Editorial facet Editorial part of music contributes to the variations among versions of a single music work. Editorial part mainly includes fingering, ornaments and dynamics. Different music performers play the same music work with different finger and hand positions, which is the fingering information. Modification is made on music to make it more beautiful or effective or to demonstrate the abilities of the interpreter [8]. Examples of ornaments are mordent and appoggiatura. A mordent is a rapid alternation from indicated note to the note above and back to the indicated note. The word "note" is used in the context of seven-note system. An appoggiatura adds a short pause in the middle of a musical note. Dynamics refers to the relative change of sound volume over the music play. A paragraph of music may be more emphasized and louder in one version than another. Editorial information may indicate the culture background of a music play but I consider it too detailed for music recommendation.

Textual facet Textual information of music or simply say lyrics is another facet of music information. Attitude and idea hide behind the semantic information of a piece of music. One may like songs that glorify sportsmanship or warriorship, whereas another prefers to listen to something with rebellious mind. Such information is reflected in lyrics so that textual information of music is also a potential source for music recommendation.

Bibliography facet Bibliographic information of music lies out of the content of audio signal, which is commonly called music metadata in some publications [14, 33, 34]. It includes song title, composer, release date, social tags and so on. Among them, social tags are most studied for music recommendation [33, 34]. The main advantage of metadata is its compactness compared with features extracted from audio. The disadvantage is that it requires data integrity and heavy load of

pre-processing work including manual work (manual tagging).

2.4 MFCC

The timbre modulation in the synthetization side of audio processing was briefly introduced in 2.2.3, but for the analysis or recognition side, timbre feature is described with different techniques. Among those timbre describing techniques, MFCC is the most popular one. This section introduces MFCC in details.

MFCC is an abbreviation of Mel Frequency Cepstral Coefficient. MFCC extraction of a piano audio signal of 43 seconds is used as an example to introduce the procedures of MFCC extraction. Figure 2.5 shows an example of an original piano signal.

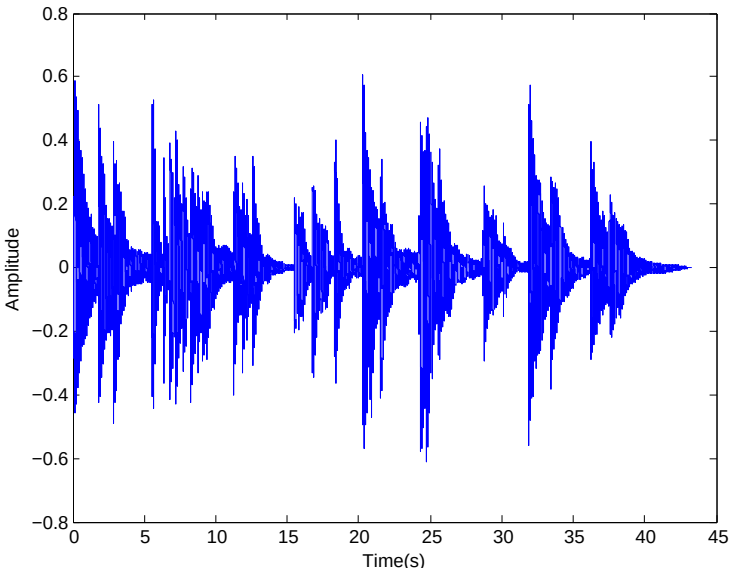

Figure 2.5: An example of a piano signal.

2.4.1 Frames

A frame is a sequence of audio signal samples in a time window. In order to avoid sharp edges at start and end of a frame, a smoothing window is commonly imposed on each frame before spectral analysis. Figure 2.6 illustrates 3 consecutive frames from the 43-second piano signal. The length of a frame is 100 ms and every frame has a 50 ms overlap with the previous one.

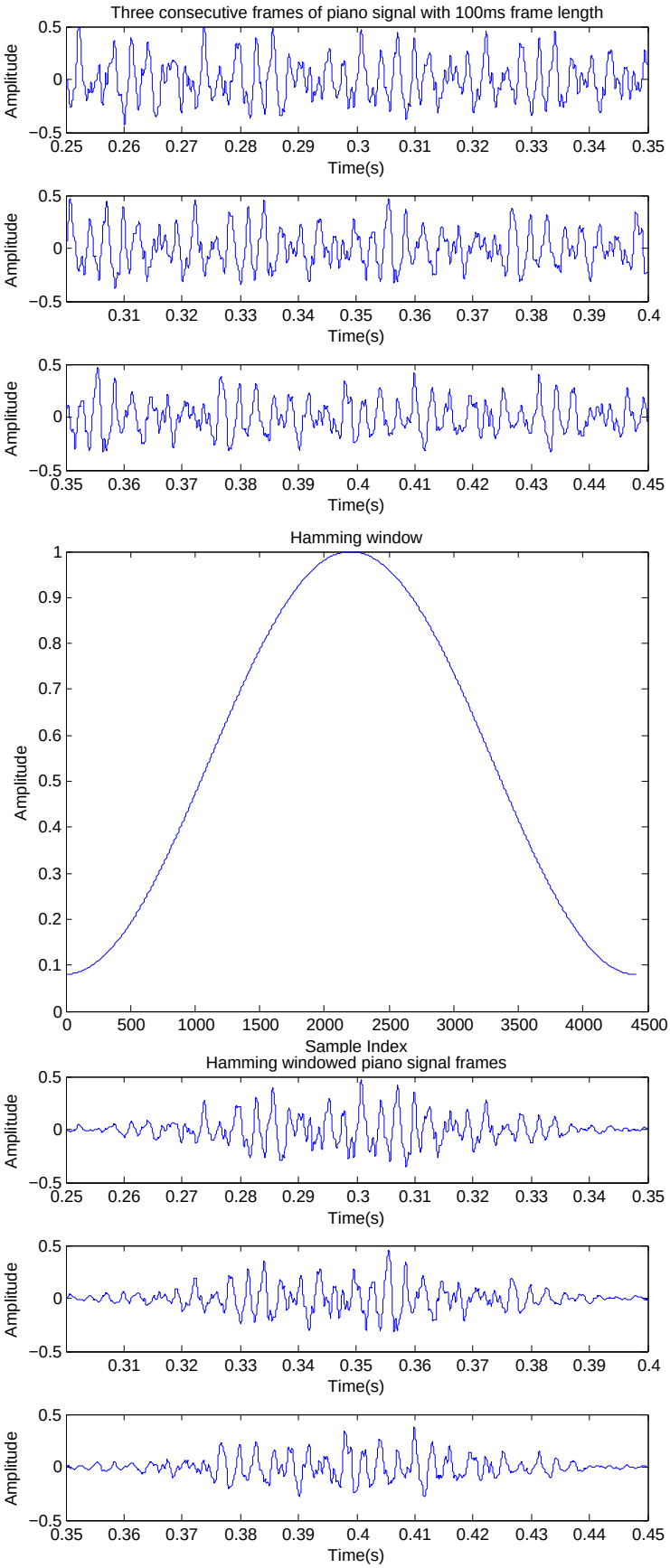

Figure 2.6: 100 ms length with 50 ms overlap, a Hamming window and Hamming windowed frames.

2.4.2 Energy Spectral Density

Energy spectral density is also called power spectrum, which represents energy on a certain frequency. It is obtained by taking the DFT (Discrete Fourier Transform) of windowed frames and calculating the square of absolute DFT value. The DFT is calculated as

$$X(k) = \sum_{n=1}^{N} x(n) e^{-j2\pi kn} \quad 1 \leq k \leq K, \tag{2.2}$$

where n represents the index of a sample ranging over the length N of a frame. K is the length of the DFT. $x(n)$ and $X(k)$ are thus time-domain amplitude of nth sample and frequency domain amplitude of kth sinusoidal component, respectively. Energy spectral density $P(k)$ of the kth component is calculated as

$$P(k) = |X(k)|^2. \tag{2.3}$$

Figure 2.7 shows energy spectral density of frames shown in Figure 2.6.

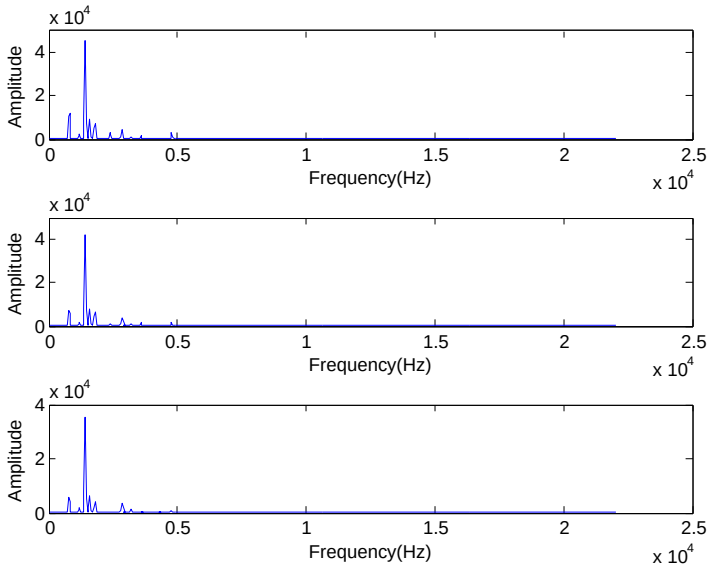

Figure 2.7: Energy spectral densities of three consecutive frames of piano signal.

2.4.3 Mel Scale

Human perception of sound frequency is not linear. Humans are more sensible for changes in pitch at low frequencies than at high frequencies. The Mel-scale formula maps measured frequency to human perceived value, so that Mel-scaled frequency better emulates the cochlea of a human. The Mel-scaling function converts frequency

f in Hz to Mel scale as

$$M(f) = 1125log(1 + \frac{f}{700}). \tag{2.4}$$

A Mel-spaced filterbank is a set of triangular filters that wraps energy spectral density into filterbank energies.

Figure 2.8 shows an example of filterbank for sampling rate at 44100.

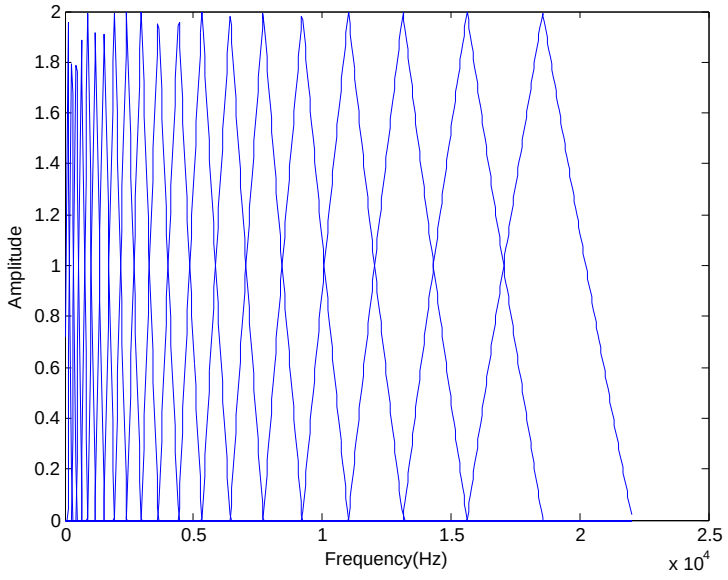

Figure 2.8: The magnitude response of a Mel-filterbank containing 20 filters.

2.4.4 Cepstrum

The name "cepstrum" was derived by reversing the first four letters of "spectrum". The short-time cepstrum was defined as the results obtained by computing power spectrum of the logarithm of the power (or amplitude) spectrum [5] in 1963. Cepstral analysis was originally used for pitch and voiced-unvoiced detection [4]. Cepstrum pitch determination is particularly effective because the effects of the vocal excitation (pitch) and vocal tract (formants) are additive in the logarithm of the power spectrum and thus clearly separate [4]. Noll writes the continuous cepstrum equation as

$$C(t) = \left|\mathcal{F}^{-1}\left\{\log(|\mathcal{F}\left\{f(t)\right\}|^2)\right\}\right|^2 \tag{2.5}$$

in [4]. In the implementation of MFCC calculation, the cepstral value is conventionally calculated by the discrete cosine transform (DCT) of a logarithm of filterbank energy values. Figure 2.9 visualizes 13-dimensional MFCC values of the example of piano signal used in this section.

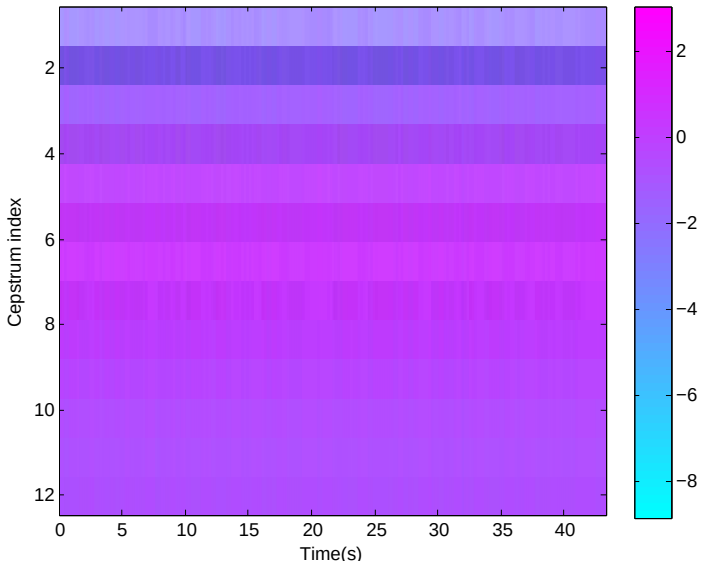

Figure 2.9: MFCC values of the 43-second piano audio signal.

2.5 Gaussian Mixture Model

Gaussian mixture model (GMM) is a probabilistic model that represents a set of data samples (either scalars or multidimensional vectors) being generated by a mixture of a finite number of Gaussian distributions. Gaussian distributions in a GMM are called mixture components. The Gaussian distribution density function of a vector $\mathbf{x}$ is calculated as

$$p_t(\mathbf{x}) = \frac{1}{(2\pi)^{(N/2)}|\mathbf{\Sigma}_t|^{(1/2)}} \exp(-\frac{1}{2}(\mathbf{x}-\mu_t)^T\mathbf{\Sigma}_t^{-1}(\mathbf{x}-\mu_t)). \tag{2.6}$$

Weights of components in a mixture model sum to one as

$$\sum_t^T \phi_t = 1. \tag{2.7}$$

Probability density function of the whole Gaussian mixture model is thus

$$p(\mathbf{x}) = \sum_t^T \phi_t p_t(\mathbf{x}). \tag{2.8}$$

Notations used in Equation (2.6)-(2.8) are specified as below:
T denotes number of mixture components .
t is the index of a mixture component.
$\phi_{i=1...T}$ denotes weights for each mixture components.
$\mu_{i=1...T}$ denotes mean values for each mixture components.

$\Sigma_{i=1...T}$ denotes the covariance matrix of each mixture components.
$p_{i=1...T}(\mathbf{x})$ denotes the probability density function of each mixture component.
$p(\mathbf{x})$ denotes the probability density function of the whole mixture model.

Covariance is a statistical measure of how variables of observations depend on each other. Given a data set that is represented by a matrix $\mathbf{X}$, the co-variance value between ith and jth dimension is calculated as

$$\Sigma_{i,j} = E[(x_i - \mu_i)(x_j - \mu_j)] = \Sigma_{j,i}, \tag{2.9}$$

where E means expectation. As is easily seen from Equation(2.9), covariance matrix is always symmetric. When every dimension of data is independent from each other, or say different features do not co-vary at all, the co-variance matrix is diagonal.

It is faster to train a GMM with diagonal covariance matrices than full covariance matrices. In practical use, some works [27, 28] use diagonal-covariance GMM to model music pieces. In [27], a diagonal-covariance GMM is used to represent music pieces and Earth Mover Distance (EMD) is used to determine music-music similarity. Examples of using full-covariance GMMs to represent music pieces are [3] and [24]. Among music similarity metrics evaluated in [24], a good result is obtained by representing music pieces with a single Gaussian with full covariance matrix and using Kullback-Leibler divergence to determine the music-music similarity.

Below is an example to explain GMM. Given a data set of body height and weight of 200 adults as is shown in the left plot of Figure 2.10, let us build a probabilistic model on it. The simplest idea is to train a two-variable Gaussian so that right plot of Figure 2.10 is obtained. The black contour plot is an equal likelihood contour of trained two-variable Gaussian distribution. Furthermore, common sense is that males and females each take about 1/2 of the population and they have different probability distribution on height and weight. With such knowledge, a mixture model with two components of equal weights $\phi_1 = \phi_2 = 0.5$ is trained as illustrated on the left plot of Figure 2.11 which uses diagonal covariance matrices. Another common sense tells me that height and weight are not independent to each other, since taller guys are more likely to be heavier. The right plot of Figure 2.11 shows the mixture model with full covariance matrices.

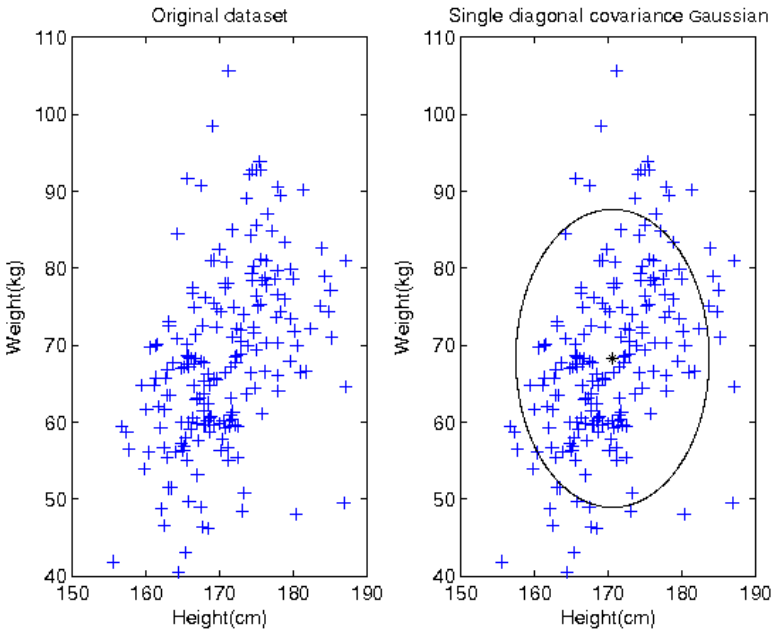

Figure 2.10: GMM example: 200 samples of weight and height data and equal likelihood contour plot of a two-variable Gaussian distribution.

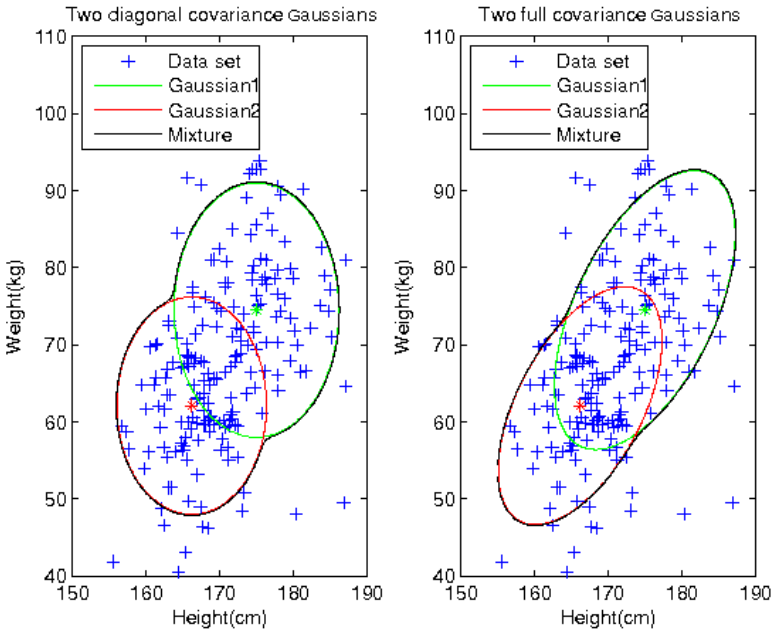

Figure 2.11: GMM example: Equal likelihood contour plots from components of mixtures and whole mixtures with of diagonal covariance matrices (left) and full covariance matrices (right).

Parameters of a GMM include mixture weights, mean of vectors and covariance matrices. They are commonly optimized through the EM algorithm that is an

iterative algorithm, which modifies parameters in every iteration to increase the likelihood of training material until the likelihood converges.

2.6 Similarity Function

As is discussed in Section 2.2, three popular music recommendation services (Pandora, Last.fm, Spotify) use similarity-based recommendation for their radio and artist-to-artist recommendation. Thus it is important to review similarity functions or distance functions applied in both collaborative-filtering and content-based methods.

In collaborative filtering, there are mainly two categories. They are memory-based and model-based. KNN (K-nearest neighbours) rating prediction is a typical memory based approach which can be traced back to as early as 1994 [21]. In the recent decade, there are many model-based methods developed, e.g. Bayesian network and latent Dirichlet allocation based models. However, KNN is still most used and studied in collaborative filtering recommendation. KNN involves similarity functions. Pearson correlation and Jaccard similarity are most used similarity functions in KNN rating prediction.

2.6.1 Pearson Correlation

In [21], the Pearson correlation function is used to calculate user-user similarity score and then the similarity score is used to predict user-item rating value. The Pearson correlation is applied on a rating matrix and can be used to calculate both user-user similarity and item-item similarity. The Pearson correlation value between two users x and y is calculated as

$$sim(x,y) = \frac{\sum\limits_{i \in I_{xy}} (r_{x,i} - \bar{r}_x)(r_{y,i} - \bar{r}_y)}{\sqrt{\sum\limits_{j \in I_{xy}} (r_{x,j} - \bar{r}_x)^2 \sum\limits_{k \in I_{xy}} (r_{y,k} - \bar{r}_y)^2}}, \tag{2.10}$$

where I_{xy} is the set of items that rated by both x and y. $r_{x,i}$ is the rating value of user x on item i. j and k are used similarly. This technique is easy to implement and easy to understand, so that it is the most popular technique. It is commonly introduced in machine learning textbooks and courses. Matrix factorization methods such as SVD (singular value decomposition) might be applied to reduce the computational complexity when the rating matrix is large. If the number of items is denoted as m and number of users is denoted as n, it takes $O(m \times n^2)$ to generate the whole similarity matrix.

In a KNN approach, an user-item rating is predicted as

$$\hat{r}_{u,i} = \bar{r}_u + \frac{\sum_{u' \in U} sim(u, u')(r_{u',i} - \bar{r}_{u'})}{\sum_{u' \in U} sim(u, u')}, \tag{2.11}$$

where U is a neighbourhood of K users that are most similar to user u. In Equation (2.11), user-user similarity is used but it is also possible to use item-item similarity.

2.6.2 Jaccard Similarity

Another important similarity for collaborative filtering is the Jaccard similarity. This similarity is originally a similarity measure for two sample sets. The Jaccard similarity between two sets is defined by the ratio of the cardinality of their intersection to the cardinality of their union:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}, \tag{2.12}$$

where A and B are two sample sets. In collaborative filtering, the Jaccard similarity between two music pieces A and B is defined by the number of users who rated both music pieces divided by number of users who rated either A or B. Rating prediction can be furthermore calculated from Equation (2.11). The time complexity of the Jaccard similarity is the same to the Pearson correlation.

2.6.3 Cosine Similarity

Cosine similarity is cosine of the angle between two vectors. Cosine similarity can be used on both ratings or item feature vectors. In [35], cosine similarity is used to calculate tag-to-tag similarities and user-to-user similarities in its social ranking algorithm. Cosine similarity between two vector represented-items or users $\mathbf{x}$ and $\mathbf{y}$ are calculated as

$$sim(x, y) = cos(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x} \cdot \mathbf{y}}{||\mathbf{x}|| \times ||\mathbf{y}||} = \frac{\sum_{i \in I} x_i y_i}{\sqrt{\sum_{i \in I} x_i^2} \sqrt{\sum_{i \in I} y_i^2}}, \tag{2.13}$$

where I is the dimensionality of vectors and x_i, y_i is the ith element of vector x and y. If x and y are two users in collaborative filtering system, their similarity is the cosine of the angle between their sums of vectors of all items they both rate. If $\mathbf{x}$ and $\mathbf{y}$ are two items in a content-based system, the cosine similarity is the cosine of the angle between their feature vectors.

2.6.4 Kullback–Leibler Divergence and Earth Mover Distance

Dmitry in [24] discussed several content-based music similarity metrics. Among them, one timbre-based metric is the Kullback–Leibler (KL) divergence based on single-Gaussian MFCC modeling. KL divergence is a non-symmetric measure of difference between two probability distributions. The KL divergence between two GMMs is not analytically tractable, nor does any efficient computational algorithm exist [31]. Dmitry uses a single Gaussian with a full covariance to model each music piece [24] . In [27], Earth mover distance is used to determine the distance between GMMs from KL divergence of every single Gaussian pairs between two GMMs [25].

The KL divergence between two univariate probability distributions is calculated as

$$D_{KL}(P\|Q) = \int_{-\infty}^{\infty} \ln\left(\frac{p(x)}{q(x)}\right) p(x) \tag{2.14}$$

$$= \int_{-\infty}^{\infty} \ln(p(x))p(x) - \int_{-\infty}^{\infty} \ln(q(x))p(x) \tag{2.15}$$

$$= -\operatorname{E}(\ln q(x)) + \operatorname{E}(\ln p(x)), \tag{2.16}$$

where P and Q are two probability distributions and x is a variable. Equation (2.15)-(2.16) are two forms of the KL divergence. Equation (2.16) is called the closed form of KL-divergence, where E stands for expectation of a probability density function. In the special case of multivariate normal distributions, the KL divergence is calculated as

$$D_{KL}(P\|Q) = \frac{1}{2}\left(Tr\left(\Sigma_P^{-1}\Sigma_Q\right) + (\mu_Q - \mu_P)^\top \Sigma_Q^{-1}(\mu_Q - \mu_P) - k - \log\left(\frac{\det \Sigma_P}{\det \Sigma_Q}\right)\right), \tag{2.17}$$

where Tr means the trace of a matrix and ^{T} means transpose of matrix. Σ_P and Σ_Q denote the covariance matrices of multivariate normal distributions P and Q. μ_P and μ_Q denote the means of P and Q. In [24], a symmetric music-music similarity is approximated as:

$$d(P,Q) = 2(D_{KL}(P\|Q) + D_{KL}(Q\|P)) \tag{2.18}$$

$$= Tr(\Sigma_P^{-1}\Sigma_Q) + Tr(\Sigma_Q^{-1}\Sigma_P) + Tr((\Sigma_P^{-1} + \Sigma_Q^{-1})(\mu_P - \mu_Q)(\mu_p - \mu_Q)^T) - 2N_{MFCC} \tag{2.19}$$

where N_{MFCC} is the number of MFCCs, that is, the dimensionality of feature vectors.

Earth Mover Distance (EMD) can be used to evaluate similarity from two set of

distances. EMD is proposed to determine music-music distances based on GMMs in [27]. N_{MFCC} term in Equation (2.19) is removed since it is constant if the system extracts features from all music pieces in the same way. Thus, the distance between two components p_i and q_j in mixture P and Q is calculated as

$$d_{p_i,q_j} = Tr(\frac{\Sigma_{p_i}}{\Sigma_{q_j}}) + Tr(\frac{\Sigma_{q_j}}{\Sigma_{p_i}}) + Tr((\mu_{p_i} - \mu_{q_j})^2(\frac{1}{\Sigma_{p_i}} + \frac{1}{\Sigma_{q_j}})). \tag{2.20}$$

A set of coefficients $f_{p_i,q_j} \geq 0$ are calculated to minimize a cost function

$$W = \sum_{i=1}^{M}\sum_{j=1}^{N} d_{p_i,q_j} f_{p_i,q_j}, \tag{2.21}$$

with following constrains:

$$f_{ij} \geq 0 \tag{2.22}$$

$$\sum_{j=1}^{N} f_{ij} \leq w_{p_i} \tag{2.23}$$

$$\sum_{i=1}^{M} f_{ij} \leq w_{q_j} \tag{2.24}$$

$$\sum_{i=1}^{M}\sum_{j=1}^{N} f_{ij} = min(\sum_{i=1}^{M} w_{p_i}, \sum_{j=1}^{N} w_{q_j}) \tag{2.25}$$

where w_{p_i} and w_{q_j} are component weights of p_i and q_j respectively. The coefficients f_{ij} are called flows and the cost function W is called work. With an optimized flow, the Earth Mover Distance is defined by the work value normalized by the sum of flows as

$$EMD(P,Q) = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N} d_{p_i,q_j} f_{p_i,q_j}}{\sum_{i=1}^{M}\sum_{j=1}^{N} f_{p_i,q_j}}. \tag{2.26}$$

3. METHOD

In this chapter, the method of my recommendation system is introduced. Background techniques such as Gaussian Mixture Model and MFCC have been introduced in Chapter 2. The fundamental thinking of my recommendation system is discussed in Section 3.1. An overview of my recommendation system is made in Section 3.2. Details of components of my recommendation system is introduced in Section 3.3-3.7.

3.1 Fundamental Hypothesis

Some similarity metrics that are used in automatic recommendation systems have been introduced in Section 2. Most of the studies about automatic recommendation are made on similarity-based techniques. However, when a real human recommends, individuals make recommendations in their own unique way and similarity-based recommendation does not seem to be overwhelmingly popular. Taking my father as an example, I asked him how he would recommend music for me. The answer was that he would recommend alternative and progressive music since I had an exploring personality. This is apparently not a similarity-based recommendation method since it does not even take into account what music I have listened to. My father's method is based on the hypothesis that types of personality are linked with styles of music. Collaborative filtering similarity-based recommendation is generally based on the hypothesis that music pieces that are liked by similar groups of users share similar preference from other users. This hypothesis uses empirical evidence to make predictions so that it seems to be reliable. Content and similarity based recommendation suggests that the similarity in music content leads to similar preferences from users. The following paragraphs discuss my personal observations on commercial music and arrive at my hypothesis, upon which my recommendation method is based on.

From personal experience, I observed such a phenomenon on modern commercial music that for most artists, people are impressed by only a few pieces from them. It is very common that a popular music consumer can name two representative music pieces for 20 artists whereas it is hard to find one who can remember all records from a single artist. Music pieces from the same artist, especially the same album, are supposed to have similar content, since they often share the same vocalist and same

set of musical instruments. However, music consumers do not often have similar preference on music from music pieces in the same album. Such a phenomenon makes the hypothesis that similar content in music leads to similar preference questionable. Thus I am not convinced by using content similarity as the only recommendation criteria. I discussed that collaborative filtering similarity-based recommendation seems to be reliable since it utilizes the empirical evidence to make prediction on user preference. How about a recommendation method that is based on probability estimation taking music content as data source? With such a question, I come up with my method and test on it in this thesis. Next two paragraphs are some of my thinking on music as a basis of my method.

Music is a form of art with sound and silence as medium. As an art form, the appreciation of music is rather complicated. How good is a piece of music? The answer could be either objective or subjective. Evidences can be easily found for both side. Beethoven is, by all classes of all nations, regarded as a great composer, which shows the objective side of music preference. However, heavy metal is loved by some people but is considered to be nothing but disturbing by some others. That is the subjective side of music preference. What aspects of information makes the evaluation of music objective, and what makes it subjective?

I illustrate my point of view with an example. “Canon in D Major” is a well-know piece of classical music and is performed solely with many instruments such as pianos, violins and electric guitars. I can see a large variance on the preferences of editions among different listeners whereas I found very low variance on the preference of the melody (temporal information is included with the term melody in this paragraph) since I found no one who simply dislikes “Canon in D Major”. Here I propose a hypothesis that the preference on melody does not vary much among individuals whereas the preference on timbres vary greatly among individuals.

With such a hypothesis, my method estimates the subjective taste of timbres from users and furthermore uses the estimated user-timbre preference to estimate the probability that a specific user accepts a piece of music. Another probability of an acceptance is estimated objectively for all users by the relative frequency that a piece of music is accepted. Two estimated probabilities are combined to give a score for each user on each music piece, by which music pieces are ranked for personalized recommendation. Detailed operations are introduced in following sections in this chapter.

3.2 System Overview

Section 3.1 introduced the basic idea of my recommendation system and this section introduces the basic structure of it. Figure 3.1 shows the flowchart of the system. As is seen in the flowchart, there are two inputs. They are music audios and records

of ratings or listening history. Features are extracted from audio signals through Echo Nest API (Section 3.3.1). A generic GMM is trained from a large amount of features (ideally all feature vectors extracted). With the generic GMM, every piece of music is represented by a timbre weight vector (Section 3.3.2 and 3.3.3). The binarizer is used to handle different data sources, the output of the binarizer is called binary rating data (Section 3.4). User timbre preference is estimated based on their binary ratings and timbre weights of music pieces that they rated (Section 3.5). For not-rated music pieces, their likelihood of being accepted is based on their timbre weights and the user-timbre preference of the user. Another probability estimation of an acceptance is calculated by the relative frequency of the music being accepted (Section 3.6). The two estimated probabilities are combined and the combined value is called utility score (Section 3.7).

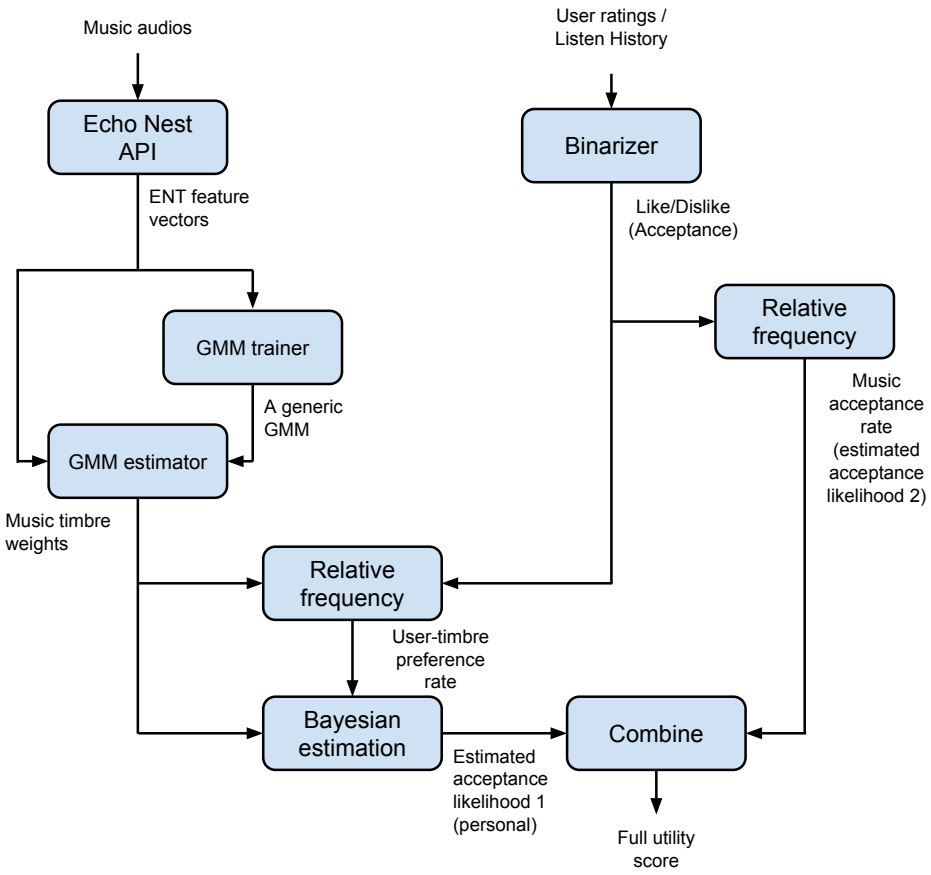

Figure 3.1: Data flow chart of my recommendation system.

3.3 Music Timbre Weight

As is introduced in the 2.1.1, hybrid recommendation systems use rating history (or listening history) and item features as recommendation sources. In this hybrid

recommendation system, the timbre weight of music is used as an item feature. In a broad sense, feature extraction transforms original input data into a reduced representation based on its nature. In this system, the audio signal of a music piece is transformed to a 64-dimensional weight vector. The process involves three steps as is shown in Figure 3.1: timbre feature extraction (Echo Nest timbre), training a generic Gaussian model and calculating probabilistic alignment for feature vectors and a sum based on the bag-of-words model.

Figure 3.2 illustrates the music timbre weight values for three music pieces. Finlandia is a symphony composed by Sibelius whereas Nemo and Amaranth are two songs from band Nightwish that is a melodic metal band active at the present age. Finlandia as a typical symphony concert consists of orchestral instruments such as violins, cellos, French horns, trumpets and a harp. Nemo and Amaranth consist of a long duration of drums, female vocals, electric guitars, an electric bass and a synthesized piano. Nemo itself has a small portion of violin and environmental sound effects of thunder weather. Briefly speaking, Nemo and Amaranth use two similar sets of instruments whereas Finlandia uses a very different set of instruments. Thus, Nemo and Amaranth are expected to have similar timbre weight values and Finlandia should have a much different timbre weight vector. Figure 3.3 shows a table of some timbre weight values of above-mentioned music pieces for a comparison. From the table, we can see that Nemo and Amaranth have, in most cases, closer timbre weight value to each other except Timbre 26 (probabilistic alignment of 26th component in generic GMM). A possible explanation is that both Nemo and Finlandia contain violins whereas Amaranth does not.

3.3.1 Echo Nest Timbre

Echo Nest is a commercial music intelligence service site that provides HTTP API to access their music analyzer. A non-paid account is limited to 120 queries per minutes to the analyzer whereas a business account can query unlimited times. Echo Nest analyzer provides diverse music descriptors including timbre, pitch and loudness in dynamic time segment. Time length of each time segment is called duration. Duration of each segment typically ranges from 200 ms to 400 ms. Figure 3.4 shows an example of Echo Nest features.

As is seen in Figure 3.4, both Echo Nest pitches and timbre has 12 dimensions. For pitches, 12 dimensions represent 12 semitones. For Echo Nest timbre (ENT), it is explained in their official document that those values are high level abstractions of the spectral surface, ordered by degree of importance [16]. Musil studies human music cognition and discusses about the choice of such long frame length and low number of dimensions of ENT in [12]. Bertin-Mahieux describes Echo Nest timbre as MFCC-like features [14]. MFCCs were reviewed in the Section 2.4.

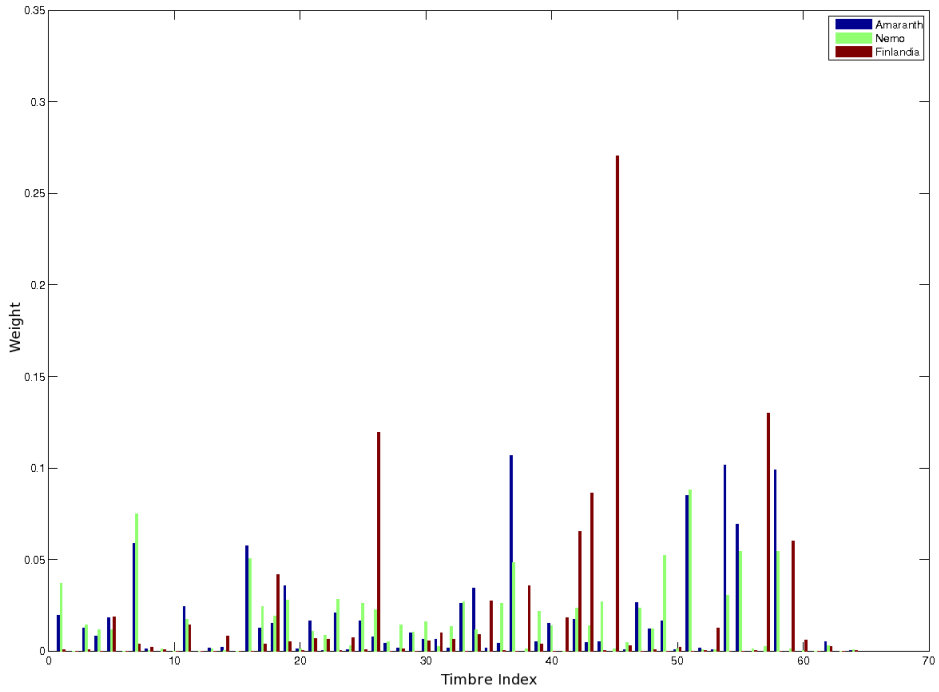

Figure 3.2: Timbre representation for Amaranth, Nemo and Finlandia.

The evaluation of my recommendation method is made using Million Song Dataset [14] and the audio features of Million Song Dataset are from Echo Nest. Thus I include Echo Nest timbre as a part of my method, however other timbre features such as MFCCs can be used as alternatives.

3.3.2 Generic GMM

With the ENT vectors calculated from Echo Nest analyzer, the next step is to establish a generic GMM with 64 timbre centroids so that timbre features are mapped to probabilistic alignments of timbre distributions. For this purpose, a generic GMM with 64 full-covariance components is trained from feature vectors with a large set of music pieces. An earlier use of a generic GMM in music recommendation is seen in [3]. The training set should include as many types of instruments and vocal styles as possible so that the model is a generic model of music timbre. Each timbre distribution in the generic model is thus generated by a frequent seen combination of music instruments and vocals. Equations (2.6)-(2.8) show how to estimate probability with a GMM.

	Timbre 3	Timbre 16	Timbre 26	Timbre 37	Timbre 45	Timbre 49	Timbre 51	Timbre 58
Amaranth	0.0587	0.0576	0.0076	0.1067	0.0001	0.0167	0.0849	0.0988
Nemo	0.0748	0.0505	0.0224	0.0485	0.0014	0.0521	0.0880	0.0545
Finlandia	0.0039	0.0000	0.1193	0.0000	0.2703	0.0000	0.0000	0.0000

Figure 3.3: Some timbre weights value for Amaranth, Nemo and Finlandia.

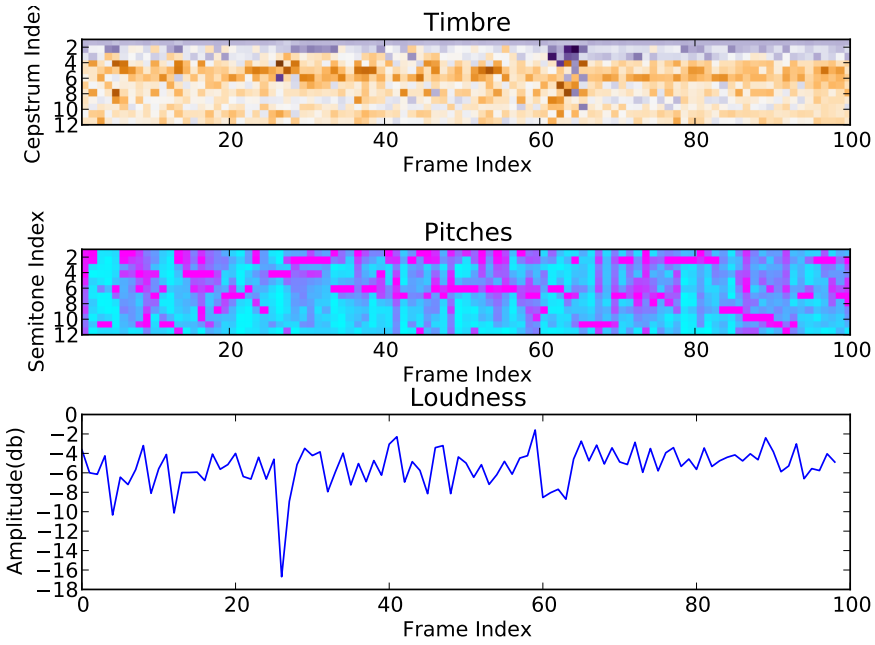

Figure 3.4: Example of Echo Nest features.

3.3.3 Bag-of-words Model

The use of the term "bag of words" can be traced back to a linguistic article [7] in 1954. The bag of words model is, originally, a simplifying representation of a document in natural language processing. With this model, a document is represented as the count of a collection of words, regardless of the positions of the words. Recently, the bag-of-words model has become a widely used model in multimedia information retrieval. The bag-of-words model is extended to information retrieval e.g. to a bag-of-words image classifier, which classifies an image by sum of classification score of each block [10].

With the bag-of-words model, a music piece is represented by a 64-dimensional timbre weights vector. A timbre weight is the mean of posterior probabilities of a timbre distribution from all feature vectors of a music piece. The timbre weights

are calculated as

$$w_{m,t} = \frac{1}{I_m} \sum_{i=1}^{I_m} \frac{p_t(f(m,i))}{p(f(m,i))}, \tag{3.1}$$

where $w_{m,t}$ is the weight of tth component (timbre distribution) in music piece m and I_m is the number of frames for music piece m. The ith feature vector of music piece m is denoted as $f(m,i)$. The estimated probabilities from tth Gaussian and whole GMM are represented with $p_t(f(m,i))$ and $p(f(m,i))$, respectively.

3.4 Data Binarization

As is shown in Figure 3.1, rating or listening history data is binarized. The results of binarization is called binary rating data in this thesis no matter if it is derived from rating data or listening history data. The demonstration of my system uses binary scale rating whereas the system evaluation is based on Million Song Dataset, which uses listening history. Binarization makes my method working for both data sources. Here we define $r_{u,m} = 1$ as acceptance and $r_{u,m} = 0$ as reject in the binary rating scale. If there is no information available of user u on misc piece m, the value is defined to be null $r_{u,m} = null$.

3.5 User-timbre Preference

User-timbre acceptance probabilities are used as parameters to estimate the probability of a user to accept a music piece. An estimated user-timbre acceptance probability is called user-timbre preference for simplicity.

3.5.1 Parameter Estimation

The goal of this step is to estimate user-timbre preference with empirical probabilities. The empirical probability or relative frequency of a user to like a timbre is the ratio of the sum of timbre weights in music pieces that the user accepted to the sum of timbre weights in music pieces that user has listened. The estimated statistical parameter of user u likes timbre t is denoted as $q_{u,t}$. This parameter is estimated as:

$$s^+_{u,t} = \sum_{\{m|r_{u,m}=1\}} w_{m,t} \tag{3.2}$$

$$s_{u,t} = \sum_{\{m|r_{u,m}\neq null\}} w_{m,t} \tag{3.3}$$

$$q_{u,t} = \frac{s+_{u,t}}{s_{u,t}}. \tag{3.4}$$

Here $s+$ and s are two sums. $s^+_{u,t}$ is the sum of timbre weights for every piece of music that user u accepted. $s_{u,t}$ is the sum of timbre weights for every piece of music that there exists a binary rating on it from the user u. In another word, $s_{u,t}$ is the sum of timbre weights for every piece of music that user u has listened.

Figure 4.6 shows an example with four users and four pieces of music. Music audios are modeled with three distributions of timbre. Let us take the preference of user 3 on timbre 2 as an example. User 3 has listened to three pieces of music. Their binary rating values are shown $r_{3,1} = 0$, $r_{3,2} = 0$, $r_{3,3} = null$ and $r_{3,4} = 1$ in the red frame in Figure 3.5. User 3 accepted only music 4 whose weight on timbre 2 is 0.1 so that the positive timbre sum is $s^+_{3,2} = w_{4,2} = 0.1$. Along with positive rate on music 4, user 3 disliked music 1 and music 2 so that $s_{3,2} = w_{1,2} + w_{2,2} + w_{4,2} = 0.3 + 0.5 + 0.1 = 0.9$. The estimated user-timbre preference of user 3 on timbre 2 is therefore $q_{3,2} = s^+_{3,2}/s_{3,2} = 0.1/0.9 = 0.111$.

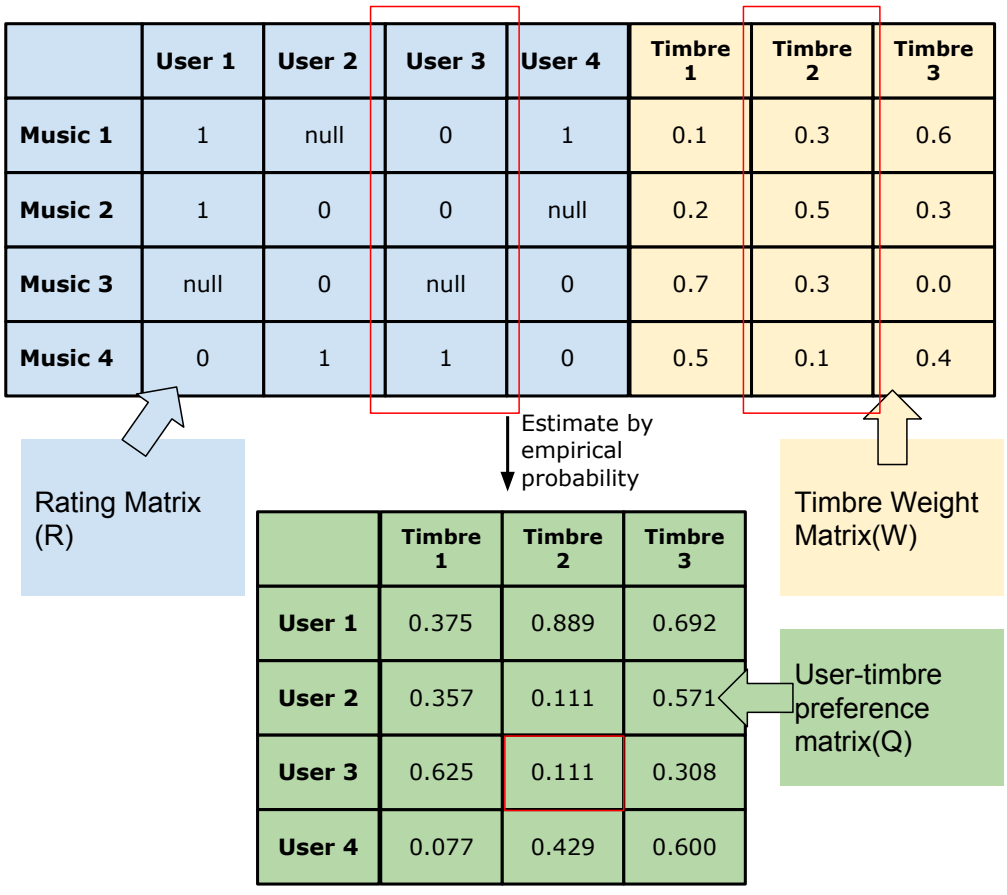

Figure 3.5: User-timbre preference estimated by binary ratings and timbre weights.

From the example above, we are going to derive the matrix form of parameter estimation. Rating matrix is defined so that each row corresponds to a piece of music and each column corresponds to a user. An element $r_{u,m}$ in the rating matrix is the

binary rating value (or $r_{m,u} = null$ when the rating is missing) for the corresponding user u and music m. The rating matrix is denoted as R. Two matrices are generated from rating matrix: a positive rating matrix and an existing rating matrix. Positive rating matrix R_p is obtained by filling missing values of rating matrix with zeros. Existing rating matrix R_e is obtained by setting all existing value in rating matrix with one and fill missing values with zeros. Following this rule, R_p and R_e that are derived from the rating matrix in above example is shown below. W stands for timbre weight matrix, where rows correspond to music pieces and columns respond to timbres. An element in the timbre weight matrix is the timbre weight value $w_{m,t}$.

$$\mathbf{R_p} = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{bmatrix} \qquad \mathbf{R_e} = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \qquad \mathbf{W} = \begin{bmatrix} 0.1 & 0.3 & 0.6 \\ 0.2 & 0.5 & 0.3 \\ 0.7 & 0.3 & 0.0 \\ 0.5 & 0.1 & 0.4 \end{bmatrix}$$

Given above three matrices, the user-timbre preference matrix Q is estimated through

$$\mathbf{Q} = (\mathbf{R_p}^T\mathbf{W})./(\mathbf{R_e}^T\mathbf{W}), \tag{3.5}$$

where ^{T} stands for transpose and ./ means element-by-element division. Elements in the timbre preference matrix are the estimated user-timbre acceptance probability $q_{u,t}$.

When a user listened to music pieces with only a narrow range of timbres, a problem is seen for user-timbre preference estimation. For example a user liked nine violin solos and disliked one violin solo without rating on any other type of music. As a result, the timbre weight sum of other timbres would be very small random-like value so that the estimated timbre preference would be inaccurate. To address this problem, users are assumed to have universal musical experience with all types of timbre. The method is to make padding with average timbre weight to smooth the user-timbre acceptance probability estimation. Average timbre weight $w_{\overline{M},t}$ is mean of timbre weight vectors of whole dataset of music as follows:

$$w_{\overline{M},t} = \sum_{m \in M} w_{m,t}/|M|, \tag{3.6}$$

where M is set of all music pieces in the system.

Another benefit to use the padding is to get different timbre preference values for those users who have only positive or only negative ratings. In practice, some users are more inclined to give positive ratings or vice versa. The extreme situation of inclined rating is that some users always like and never dislike, which makes all user-timbre preference estimated to 1, making the parameters non-sense. Padding

operations is done by

$$\tilde{s}^{+}_{u,t} = s^{+}_{u,t} + \alpha w_{\overline{M},t} \tag{3.7}$$

$$\tilde{s}_{u,t} = s_{u,t} + \beta w_{\overline{M},t}, \tag{3.8}$$

where $\tilde{s}$ means padded timbre weight sum. Two padding factors α and β are multiplied to average timbre weight. The default user-timbre preference probability value when there is no evidence on a timbre is determined by $\alpha/\beta < 1$. The higher this ratio is, the less smooth comparison between high preferable and low preferable timbres.

3.5.2 Acceptance Probability Prediction

The conditional probability that a user accepts a music can be formed from a generative model through the Bayes' rule, with timbres as a hidden variables:

$$p(r_{u,m} = 1) = \sum_{t \in T} p_m(t) p(r_{u,m} = 1|t), \tag{3.9}$$

where $p(r_{u,m} = 1)$ stands for the probability of user u accepts music piece m. The probability that music m generates timbre t is $p_m(t)$. The probability that a user accepts a timbre is assumed to be always the same no matter what music the user listens to so that for the same u and t, $p(r_{u,m} = 1|t)$ is independent of m. Let us use $\hat{r}_{u,m}$ to denote the estimated probability that user u likes music m with a set of T timbres as hidden variables. The probability is estimated as

$$\hat{r}_{u,m} = \sum_{t \in T} w_{m,t} q_{u,t}, \tag{3.10}$$

where $p_m(t)$ is estimated with the timbre weight $w_{m,t}$ discussed in Section 3.2 and the probability that a user likes a timbre is estimated with user-timbre preference $q_{u,t}$ discussed in Section 3.4.1. The matrix form of this process is as

$$\hat{\mathbf{R}} = \mathbf{Q}\mathbf{W}^T \tag{3.11}$$

where $\hat{\mathbf{R}}$ is the matrix of estimated probability $\hat{r}_{u,m}$. $\mathbf{Q}$ and $\mathbf{W}$ are user-timbre preference matrix and timbre weight matrix respectively.

Figure 3.6 continues the example used in Figure 3.5. It takes the user-preference matrix obtained and the timbre weight matrix used in Figure 3.5 as input to predict the probability of an acceptance.

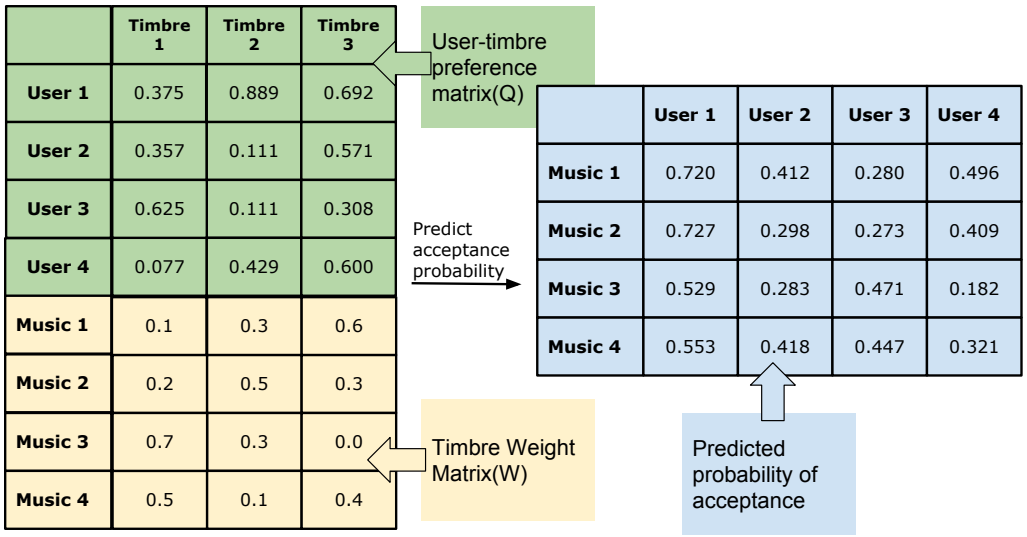

Figure 3.6: Predicted probability of acceptance by user-timbre preference.

3.6 Music Acceptance Rate

Acceptance probability prediction from merely the user-timbre preference is not comprehensive, since there are many other aspects of music that affect the user acceptance. Instead of trying to estimate the effect of other aspects on the preference of an individual user, the acceptance rate of music (relative frequency of positive ratings) is used as a supplement to the estimation with user-timbre preference. Music acceptance rate is a simple probability estimation of all users to accept a music piece from empirical probability. For example, two piano solo pieces may have very similar timbre weights but one is a master piece whereas the other is played by an amateur. The acceptance rate can help making a difference in such a condition. Music acceptance rate is calculated as

$$\overline{r}_m = \frac{|\{u|r_{u,m} = 1\}|}{|\{u|r_{u,m} \neq null\}|}, \tag{3.12}$$

where $|\{u|r_{u,m} = 1\}|$ is the number of users who accepted music m and $|\{u|r_{u,m} \neq null\}|$ is the number of users that there exists ratings from them on music m. The matrix of music acceptance rate is calculated as

$$\overline{\mathbf{R}} = [(\mathbf{R_p} \cdot \mathbf{1})./(\mathbf{R_e} \cdot \mathbf{1})] \cdot \mathbf{1}^T, \tag{3.13}$$

where $\overline{\mathbf{R}}$ has every column the same and the mth element of every column is $\overline{r}_m$. $\mathbf{R_p}$ and $\mathbf{R_e}$ have been defined in Section 3.4.1. $\mathbf{1}$ represents an all-one column vector with the length of the column numbers of the matrix that it multiplies and ./ represents matrix element-wise division. Figure 3.7 shows the operation with above example.

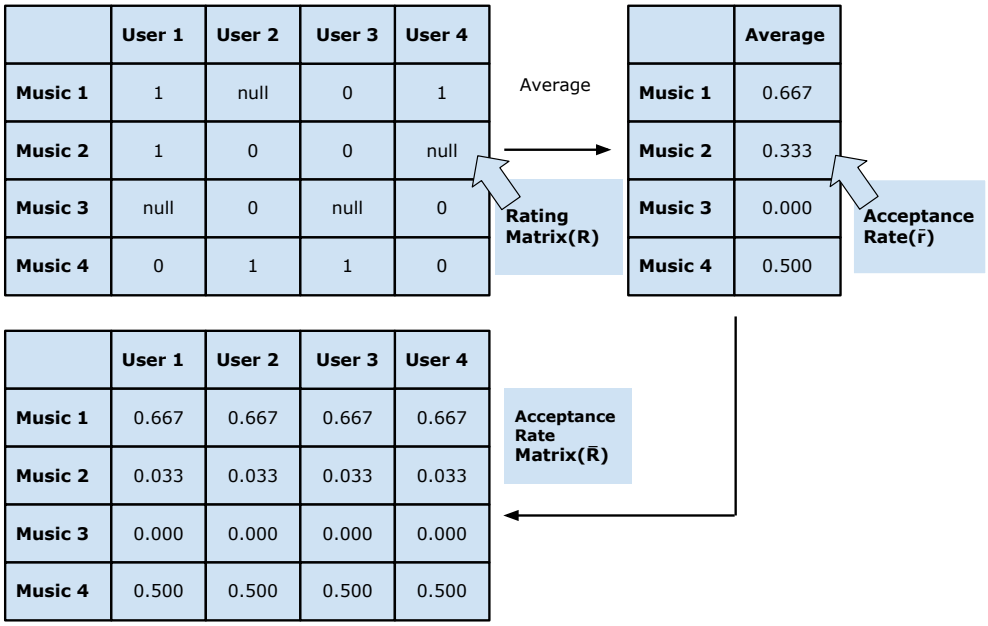

Figure 3.7: Calculation of music acceptance rate matrix.

A low number of ratings on a music piece may lead to an extreme estimation, as music 3 in the example. The discounting method (Good-Turing smoothing) [36] is used to deal with over-estimation of low-counted words in trigram model in natural language processing. Similarly, a smoothed counting is adopted in this work for a practical reason. The smoothing method assumes that every piece of music is accepted by at least one person and rejected by one person in the world. In another words, there is no piece of music of 0 or 100 percent probability to be accepted with the smoothing estimation. The smoothing method used in my system is as:

$$\overline{r}_m = \frac{|\{u|r_{u,m} = 1\}| + 1}{|\{u|r_{u,m} \neq null\}| + 2}. \tag{3.14}$$

The smoothing method makes padding with one extra positive rating and one extra negative rating. One advantage of such discounting is that estimated probability value can be given value 0.5 when no rating data is available. Another advantage is that for music that has all positive or negative ratings, the estimated acceptance probabilities are distinguished by the number of ratings. For instance, music A gains

5 positive ratings without negative rating and music B gains 10 positive ratings and also no negative rating. In this case, modified average rating of B is $11/12 = 0.917$ higher than that of A, $6/7 = 0.857$.

3.7 Combine and Rank

We have estimated the probability that a user accepts a music piece in two ways. The first estimation uses timbres as hidden variables and the second one simply uses the music acceptance rate. Now we need to decide a function to combine these two estimated probability so that music pieces are ranked to users. The function value is called utility score or full utility score in this thesis. An requirement for the combination function is that it should increase along with both estimated probabilities. In the evaluation, weighted arithmetic mean

$$f_{u,m} = \frac{w_1 \hat{r}_{u,m} + w_2 \overline{r}_m}{w_1 + w_2}, \tag{3.15}$$

weighted geometric mean

$$f_{u,m} = \sqrt[w_1+w_2]{(\hat{r}_{u,m})^{w_1} (\overline{r}_m)^{w_2}}, \tag{3.16}$$

and weighted harmonic mean

$$f_{u,m} = \frac{w_1 + w_2}{w_1/\hat{r}_{u,m} + w_2/\overline{r}_m}, \tag{3.17}$$

are tested.

To avoid giving the same recommendation playlist when a user repeats recommendation requests, a recommendation pool is used. A recommendation pool consists of top ranked music pieces for a user. When a user requests recommendation, a subset is randomly selected from recommendation pool and responded as a playlist. The size of recommendation pool should be much larger than playlist. For example, 10 pieces of music are recommended from a pool of top 100 ranked music pieces.

4. ALGORITHM ANALYSIS

This chapter starts with the introduction on the storage management of my system running as an online service. Furthermore, this chapter discusses how the system meets common operational requirements of an automatic recommendation system. Common notations in this chapter is as below:

Symbol	Meaning
U	the number of users in system
M	the number of music pieces
T	the number of Gaussians in the generic GMM
E	the total number of ratings
K	the number of songs that the recommendation is based on

Table 4.1: Notations of sizes of recommendation systems.

4.1 Storage Management

The complexity of an algorithm is dependent on how the storage of data is managed. In both my system and item-to-item similarity based recommendation systems, there is important core data with two indexes. For my system, there are binary ratings with indexes of users and music pieces, timbre weights with indexes of music pieces and timbres and user-timbre preferences with indexes of users and timbres. For a item-to-item similarity based system, there is music-to-music similarity data with a pair of music as indexes. I consider two ways to manage these data (there is possible better ways that I do now know) . One is to use a matrix representation stored with a sequence of consecutive physical addresses so that an element in the matrix is searched in constant time. The drawback is that the data structure is not incremental for both dimensions. For a $m \times n$ matrix, it takes $O(m \times n)$ to resize to $(m + 1) \times (n + 1)$ due to the shifting of elements. The second way is a list representation managed by a database management system: in each row, two indexes are two indexed columns and the value is stored in another column. With such representation, a query costs $O(\log(mn))$ or $O(\log(m) + \log(n))$ for an element in $m \times n$ rows of records. The advantage is its incrementality since no shifting is required when the size of data enlarges.

For ratings and item-to-item similarities, matrix representation is proper to use

only when their sizes are fixed. This leads to the discussion of two options for updating the system with newly available music pieces and ratings. One is to update the system as an time-scheduled activity so that ratings and similarities from fixed number of users and music pieces can be stored in matrix representation and searched constantly. When the system updates, new matrices are constructed. Another option is to update instantly. For this purpose, ratings and similarities are needed to be managed in list representation in database management system to keep an dynamic size of the data set.

One advantage for my system is that core data in my system can be stored in a matrix representation with dynamic number of users and music pieces. The number of timbres is constant and small ($T = 64$ in my system) so that timbre weights and user-timbre preferences has one dimension with constant number. For example, when one user and one piece of music is added to the system. The size of timbre weight matrix and the user-timbre preference matrix become $(M + 1) \times T$ and $(U + 1) \times T$, respectively. On the contrast, rating matrix and music-to-music similarity matrix become $(M + 1) \times (U + 1)$ and $(M + 1) \times (M + 1)$, respectively.

There is also other information needed for my system and they are stored along with timbre weights and user-timbre preferences.The number of ratings that a user made is needed to update user-timbre preference. The number of ratings made on a music piece is needed to update music acceptance rate and music acceptance rate is needed to calculate full utility score. Music acceptance rate, the number of ratings made to a piece of music and number of ratings a user made are stored in additional columns in the two incremental matrices.

4.2 Scalability and Reactivity

Scalability is the ability to cope with a growing number of data. Scalability depends on the computation complexity of the system. Reactivity is the ability of the system to keep up with recent changes. In music recommendation, reactivity includes changing recommendation result as new ratings and newly added music pieces available. From the discussion about the two options to update a system, we can easily conclude that a system that updates instantly has better reactivity than a system that updates time-scheduled. Table 4.2 and Table 4.3 summarize complexities with the two options for my system, item-to-item similarity based collaborative filtering system and item-to-item similarity based content-based system. The highlight is that my method is computationally cheap when the system updates with newly available ratings and music pieces instantly.

Let us consider the computation cost to update a system instantly. For a collaborative filtering system, keeping the similarity matrix updated with a newly added rating is so costly that $\theta(M \times U)$ is needed to recalculate Pearson (could be pos-

sibly Cosine or Jaccard) similarity values between newly rated item and all other items. For my system, since the timbre weights and user-timbre preferences are stored in a matrix representation, a search or a update runs in constant time. Since the search for the music timbre weights and the update of user-timbre preferences is made on T timbres, the computation cost to update with a rating is $\theta(T)$. The acceptance rate of the corresponding music piece is also updated in constant time. For a content-based system, update with rating is not needed.

When a piece of music is added to my system, T timbre weights are calculated from T timbre distributions. For a collaborative filtering system, newly added pieces cannot be recommended and nothing is needed to do with it. For content-based system, the similarities between the new piece and all other pieces need to be calculated and added into database. The computation cost is $\theta(M \log(M))$.

When a user requests a recommendation, my system needs to calculates the utility score of all music pieces and select top ranked pieces. For every single piece of music, calculating a utility score costs $\theta(T)$. The computation cost for all utility scores of given the user is $\theta(T \times M)$. Finding top ranked pieces costs $O(M)$ with selection algorithm [37]. The total cost to make a recommendation with my system is thus analyzed to $\theta(T \times M)$. For an item-to-item similarity based system, the similarities are stored in a list representation in the database so that it costs $\log(M)$ to query for a piece. For finding similarities between a music piece and all other music pieces, the computation cost is $\theta(M \log(M))$. Finding top similar pieces costs $\theta(M)$ with the selection algorithm. If the system recommends top similar pieces to K pieces that a user accepts, the total cost is $\theta(K \times M \times \log(M))$.

	My system	Item-to-item CF	Item-to-item CB
Add a rating	$\theta(T)$	$\theta(M \times U)$	
Add a music piece	$\theta(T)$		$\theta(M \times \log(M))$
A recommendation	$\theta(T \times M)$	$\theta(K \times M \times \log(M))$	$\theta(K \times M \times \log(M))$

Table 4.2: Time complexity comparison between my recommendation system, item-to-item similarity based collaborative filtering systems and item-to-item similarity based content-based systems with instant updates.

For recommendation systems that update time-scheduled, the set of users and the set of music pieces in a system is fixed during a update. The output of a time-scheduled update is a cache of what music pieces should be recommended to each user, by which a recommendation can be made to a user in constant time. For my system, timbre weights, user-timbre preferences and music acceptance rates updates with every rating. Since it takes $\theta(T)$ to update with a rating, it takes $\theta(T \times E)$ to calculate timbre weights, user-timbre preferences and music acceptance rates. The matrix of utility scores is calculated as is introduced in Chapter 3 that $\theta(T \times M \times U)$ is needed (Mainly the matrix multiplication between two matrices with sizes of $M \times T$ and $T \times U$, respectively). Picking top ranked music pieces for

each user costs $O(M \times U)$. Since $E \leq M \times U$, the time complexity of my system to calculate all recommendations from user ratings is thus $\theta(T \times M \times U)$. For a collaborative filtering system, calculating similarity matrix from rating matrix costs $\theta(M^2 \times U)$ and picking top similar music pieces to K pieces that are accepted by each user costs $O(K \times M \times U)$. Since K is a small constant, the total computation cost is analyzed to $\theta(M^2 \times U)$. For a item-to-item similarity based content-based system, $\theta(M^2)$ is needed to calculate similarities between every pair of music pieces and it takes also $O(K \times M \times U)$ to pick music pieces for users. The computation cost is analyzed to $O(M^2 + K \times M \times U)$ and $\Omega(M^2)$ for a item-to-item similarity based content-based recommendation system to calculated all recommendations.

	My system	Item-to-item CF	Item-to-item CB
Time complexity	$\theta(T \times M \times U)$	$\theta(M^2 \times U)$	$\Omega(M^2)$

Table 4.3: Time complexity comparison between my recommendation system, item-to-item similarity based collaborative filtering systems and item-to-item similarity based content-based systems to calculate all recommendations as time-scheduled update.

4.3 Cold Start

In the beginning of a recommendation service when only limited rating (or listen history) data is available, the system needs to be able to provide useful recommendations. This is a big challenge for a collaborative filtering recommendation system since its recommendations are based on ratings of other users. However, in my system, ratings of other users are utilized only for obtaining the acceptance rate. If there is only one user in the system, a meaningful recommendation by pure user-timbre preference score would be made. From the evaluation result shown in the next chapter that ranking accuracy of pure user-timbre preference score is about 0.56, and after taking the acceptance rate into account, the ranking accuracy is improved to 0.59. Pure content-based recommendation suffers no cold start.

When a new music piece is added to the system, a collaborative filtering system cannot give a similarity value between new piece and another piece until one user gives rating to both of them. A content-based recommendation system does not use ratings so that there is no computation needed to deal with new ratings and a new music piece is available for recommendation.

Another thing to mention is the possibility to get a recommendation as an anonymous user. My system is a personalized one so that it needs ratings from the user itself. Songs with top acceptance rate will be recommended to anonymous users or users that have not rated yet. The cold start problem described in this section is summarized in Table 4.4.

	My system	CF	CB
Works without any rating data	✓		✓
Performance improves with more ratings	✓	✓	
Recommend newly added music	✓		✓

Table 4.4: Starting performance comparison between my recommendation system, item-to-item similarity based collaborative filtering systems and item-to-item similarity based content-based systems.

5. EVALUATION

The evaluation is conducted using Million Song Dataset, which comes from LabROSA and is used on Million Song Challenge, that is, an awarding contest held on Kaggle. The evaluation tests ranking accuracy of my system.

5.1 Dataset

The Million Song Dataset is a freely-available collection of audio features and metadata for one million contemporary popular music tracks [14]. Figure 3.4 shows an example of audio features contained in Million Song Dataset. Metadata of Million Song Dataset such as lyrics, song level tags are not used in my recommendation method. Besides audio features, play count data is also used. Some statistics of play count data in Million Song Dataset is shown in Table 5.1.

Users	1,019,318
Songs	384,546
Play counts	48,373,586

Table 5.1: Statistics of Million Song Dataset.

The user-song play counts matrix $\mathbf{H}$ is exemplified as

$$\mathbf{H} = \begin{bmatrix} 0 & 0 & 1 & 10 & 0 \\ 1 & 1 & 0 & 0 & 5 \\ 0 & 4 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 2 \\ 1 & 1 & 2 & 0 & 0 \end{bmatrix},$$

where a row corresponds to a user and a column corresponds to a song, and the value is how many times the user listen to the song.

The binarization on the play counts is done as follows: If a user listens to a song for only one time, it is considered that the user rejects that song since the user does not choose that song again. If a user listen to a song for multiple times, it is considered that the user accepts that song. After such a operation, above listen history matrix is transformed to a binary rating matrix $\mathbf{R}$ with missing values as shown below.

$$R = \begin{bmatrix} null & null & 0 & 1 & null \\ 0 & 0 & null & null & 1 \\ null & 1 & null & null & null \\ 0 & null & 0 & null & 0 \\ 0 & null & null & 0 & 1 \\ 0 & 0 & 1 & null & null \end{bmatrix}$$

It must be noted that such a transformation brings noise to the data. For example, a user might enjoy a song and listen to it often in his own music playing device but the user plays it only once on the online service. Such a noise might have negative impact on the evaluation result. Although play counts data is not an ideal source, Million Song Dataset is still chosen for the evaluation because it is the only large dataset that contains both audio features and user-music interaction data.

5.2 Ranking Accuracy

To study the performance of a top-N personalized recommender, an objective experiment is more effective than observational studies. An experiment that investigates the feedback of a randomized group of users on their top-N items is an ideal evaluation for the performance of a top-N personalized recommender. However, it costs too much labour time to collect enough amount of experimental results. The problem of an observational study on a top-N personalized recommender is the sparsity of a dataset. As is shown in Table 5.1, Million Song Dataset has 48 373 586 play counts and 1 019 318 users. On average, a user has about 48 records of play counts. If top 100 music pieces out of the total 384 546 music pieces are recommended, the expected number of recommended music pieces that has been played by a user in the test set is approximately 0.004. The observational study fails with such a small number of top-N items being tested.

To better utilize the observations in the dataset, ranking prediction is chosen to evaluate the system. Thus, all ratings from a user in the test set instead of only ratings on top-N items are evaluated. The ranking accuracy metric is calculated as follows: for every user, their ratings in the test set are compared pair wise. The pairs are divided into 3 groups: A set of contradictory pairs C^- consists of pairs where preference order between system ranking and user ranking is contradictory. A set of corporate pairs C^+ consists of pairs where preference order between system ranking and user ranking conform. Unused pairs C^u consists of pairs where either system or user ranks two items the same. The ranking accuracy is calculated as

$$Accuracy = \frac{C^+}{C^+ + C^-}. \tag{5.1}$$

It is important to notice that the ranking accuracy is not an very pertinent eval-

uation for my top-N recommendation system. Since only top-N items are recommended, the ranking of other items are in fact irrelevant to the system performance. However, all rankings are taken into account in the evaluation because of the rating sparsity discussed earlier in this section.

5.3 Results

I put two out of every three play counts from each user into the training set and put the rest into the test set. Heuristically, I found out that the ranking accuracy does not vary much when the number of users in the test set is large enough. For time saving purpose, I use only users with more than 100 ratings in the test set to make the evaluation. To test on three combination functions (arithmetic mean, geometric mean and harmonic mean) that is discussed in Section 3.7, I test different weight pairs of (w_1, w_2), where w_1 is the weight for the estimated probability based on user-timbre preference and w_2 is the weight for acceptance rate. When only acceptance rate is used, $w_1 = 0$ and when only the estimated probability based on timbre is used, $w_2 = 0$. Ranking accuracy is tested with following weight pairs: $(w_1 = 0, w_2 = 1)$, $(w_1 = 1, w_2 = 4)$, $(w_1 = 1, w_2 = 2)$, $(w_1 = 1, w_2 = 1)$, $(w_1 = 2, w_2 = 1)$, $(w_1 = 4, w_2 = 1)$, $(w_1 = 8, w_2 = 1)$ and $(w_1 = 1, w_2 = 0)$.

Function $\setminus \frac{w_1}{w_2}$	0	0.25	0.5	1	2	4	8	∞
Arithmetic mean	0.568	0.580	0.581	0.585	0.589	0.592	0.589	0.559
Geometric mean	0.568	0.580	0.582	0.585	0.589	0.591	0.585	0.559
Harmonic mean	0.568	0.580	0.582	0.585	0.588	0.589	0.586	0.559

Table 5.2: Ranking accuracy using weighted arithmetic mean, weighted geometric mean and harmonic mean as combination function. The result is ordered by relative weights of timbre-based estimated probability and acceptance rate ascending from left to right.

As is seen in the Table 5.2, three combination functions have similar performance in ranking prediction. The best performance comes from a relative weight of 4. Ranking accuracy of random recommendation and recommending by popularity is calculated for comparison.

Ranking by	Ranking Accuracy
Popularity	0.557
Random	0.502

Table 5.3: Ranking accuracy of random ranking and ranking by popularity.

5.4 Examples of recommendation results

The evaluation based on Million Song Dataset provides the statistics about the performance of my system. It is also important to investigate on the instances of recommendation results. Since Million Song Dataset provides only audio features and the artists in Million Song Dataset are mostly from U.S. and are not familiar to me, I manually collected a small dataset to demonstrate recommendation results.

The dataset contains 189 pieces of music from 40 top popular artists from Finland according to Echo Nest popularity statistics. The url of the demonstration page is http://shuyang.eu/plg. Links of all music pieces are external and some links are already broken. There are 448 ratings collected from my friends. In the demonstration, 8 top ranked pieces are ordered by utility score. The combination function is the geometric mean (not weighted) of the two estimated probabilities. The demonstration system was built before my evaluation with Million Song Dataset when I had no idea about the choice of combination function.

Table 5.4 shows statistics of some artists in the demonstration dataset. The genres of the artists are derived from Wikipedia. Music pieces from metal bands take a large proportion in the demonstration dataset but the dataset still covers music pieces with a wide range of genres.

Name of Artist	Number of pieces	Genre
Nightwish	25	symphonic power metal
Amberian Dawn	4	symphonic metal
Eternal Tears of Sorrow	4	symphonic death metal
Stratovarius	8	power metal
Sentenced	11	melodic death metal
Fintroll	15	folk metal
Apocalyptica	20	neoclassic metal
Jean Sibelius	3	classic
Anna Puu	7	pop female singer
Mari Boine	6	female folk
Mirel Wagner	2	female folk
Rubik	7	Indie pop
Satellite Stories	4	Indie pop
PMMP	6	pop rock
Luomo	5	electronic
Lordi	8	hard rock
H.I.M	4	rock

Table 5.4: Statistics of some artists in the demonstration dataset.

In Section 3.3 we have discussed the instrument compositions of Amaranth and Finlandia. Here we start the investigation from rating these two pieces. I simply give a positive rating to Finlandia and a negative rating to Amaranth. The recommendation results based on such a rating history is shown as Figure 5.1. Three of 8 recommendations are from the neoclassic metal band Apocalyptica. These three pieces mainly consists of cellos. And the other two pieces from Sibelius are also recommended. The reason that it ranks after Ruska and Beautiful is their low rate of acceptance. "Bakom varje fura", the fourth recommended piece consists of multiple wind instruments without singing. The seventh recommended piece Mourun consists of drums and an electric guitar playing single pitch melody. "Beaiveldttas Butterfly" is a piece of Shamanic music that consists of a female singer and a Shamanic drum. As a summary, because of Amaranth is disliked, timbres related to human voice are estimated to be less preferable and symphonic timbres are estimated to be more preferable so that such a recommendation results are given.

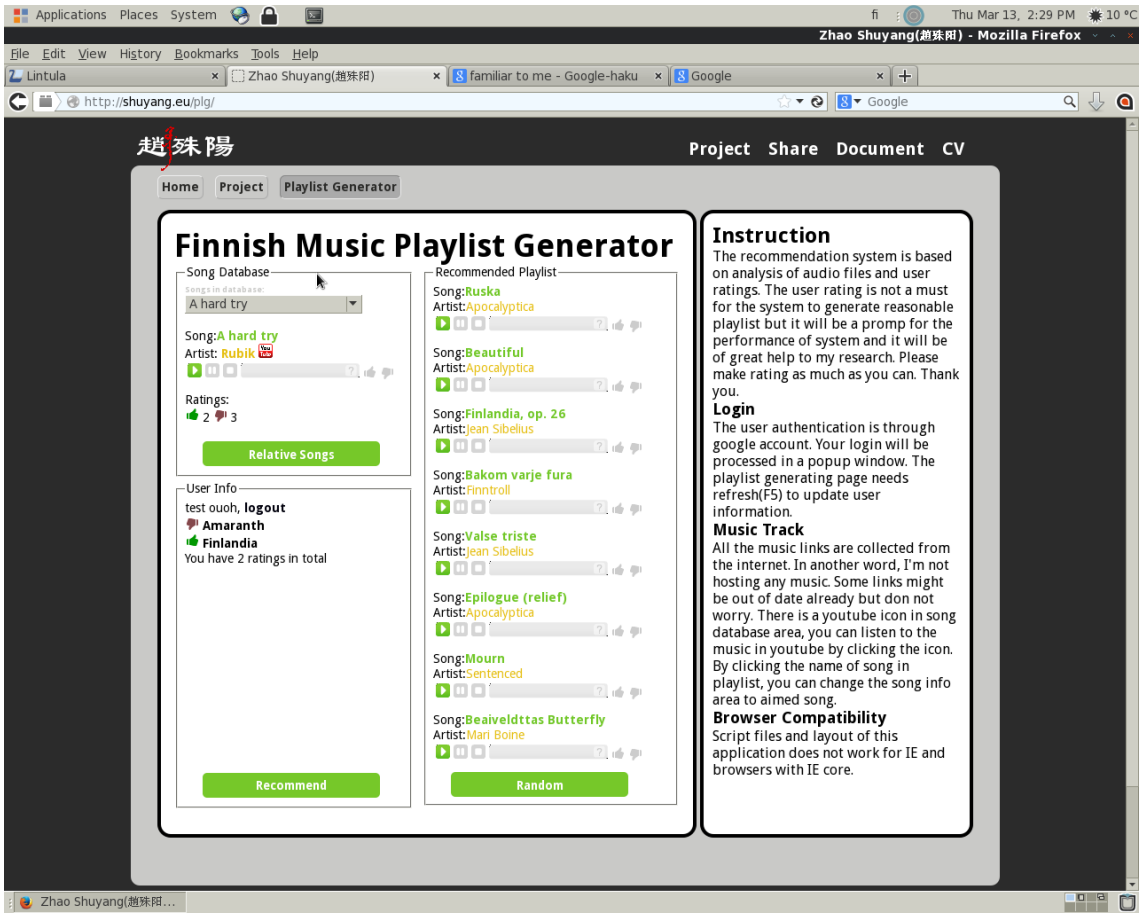

Figure 5.1: An example of recommendation results.

Now, we are going to like a piece with female vocal and check the recommendations. I added a positive rating on the female pop music "Nopeimmat junat" by Anna

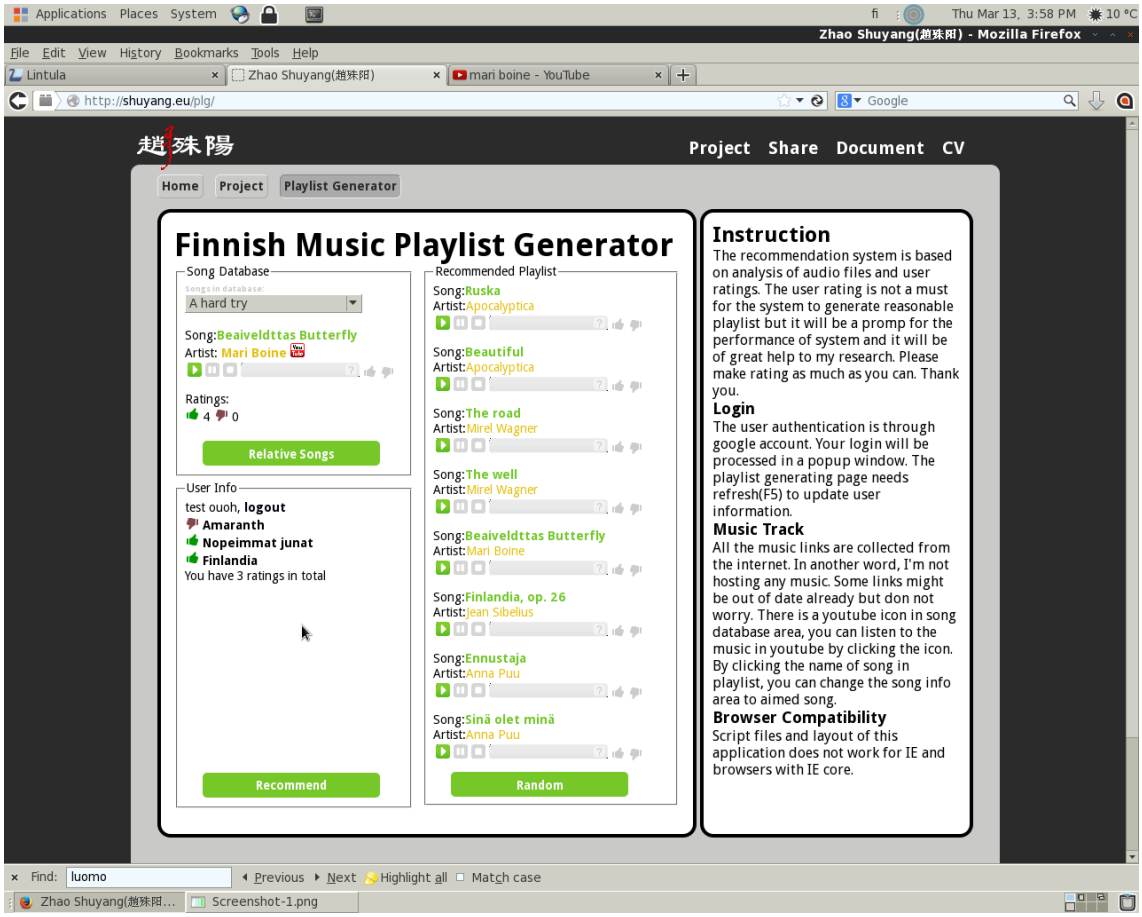

Figure 5.2: An example of recommendation results.

Puu to my rating history and the recommendations are shown in Figure 5.2. We see that first two recommendations remain the same since my preference of cellos is still estimated high. Following recommendations are music pieces that have a female vocalist and light accompaniment except Finlandia as the sixth recommendation. Mirel Wagner uses a single acoustic guitar as accompaniment and the seventh and eighth recommendations are from Anna Puu, the artist of the newly liked piece.

6. SUMMARY AND CONCLUSION

In this thesis work, a hypothesis on the consumers' preferences of modern popular music is proposed based on my personal observations that the preferences of timbral information in music vary greatly among individuals whereas the preferences of melodic information in music vary slightly among individuals.

A method of automatic music recommendation system is proposed based on my hypothesis. The recommendation system takes audio signal and rating data as input and outputs personalized recommendations. Each user gets a score for every piece of music and top ranked music pieces are selected as candidates. A random recommendation is made among candidates. The score used for ranking combines two estimated probabilities of an acceptance. One estimated probability is based on the estimation of users' preferences on timbres. Another estimated probability is the empirical probability that a piece of music is accepted.

The method addresses well traditional problems of automatic recommendation systems such as reactivity and cold start. An demonstration of my method as an online service is available at http://shuyang.eu/plg. From the investigation on the recommendations result using the demonstration dataset, the system well meet my expectation of the way it is supposed to recommend. The recommendation accuracy is evaluated based on Million Song Dataset using the criteria of pairwise ranking accuracy. Several combination functions for the two estimated probabilities are tested. The best ranking accuracy is achieved with a weighted arithmetic mean function as a combination function. Compared to the ranking accuracy of random ranking (theoretically 0.5, experimentally 0.502) and ranking by popularity (0.557), my recommendation system clearly outperforms them (0.592). Unfortunately, I have not found any other music recommendation system evaluated with ranking accuracy. Ranking accuracy is more commonly used in the evaluation of web page ranking. The famous Page Rank algorithm has an accuracy of 0.567 [38].

As a conclusion, I would say that my recommendation system is a practical choice for a real-time online music recommendation task.

BIBLIOGRAPHY

[1] B. Shao, D. Wang, T. Li, and M. Ogihara, "Music Recommendation Based on Acoustic Features and User Access Patterns," IEEE Transaction on Audio, Speech, and Language Processing, vol. 17, no. 8, Nov. 2009.

[2] K. N. Rao and V.G. Talwar, "Application Domain and Functional Classification of Recommender Systems—A Survey," Journal of Library & Information Technology, Vol. 28, No. 3, May 2008.

[3] M. Goto, K. Komatani, T. Ogata, "An Efficient Hybrid Music Recommender System Using an Incrementally Trainable Probabilistic Generative Model," IEEE Transsction on Audio, Speech, and Language Processing, vol. 16, no. 2, Feb. 2008.

[4] A. Michael Noll, "Short-Time Spectrum and Cepstrum Techniques for Vocal-Pitch Detection." Journal of the Acoustical Society of America, Vol. 36, No. 2, pp. 296-302. Feb, 1964.

[5] B. P. Bogert, M. J. R. Healy, and J. W. Tukey, "The Quefrency Alanysis of Time Series for Echoes: Cepstrum, Pseudo Autocovariance, Cross-Cepstrum and Saphe Cracking." Proceedings of the Symposium on Time Series Analysis pp. 209-243. New York: Wiley, 1963.

[6] P. Melville and V. Sindhwani, "Recommender Systems." In Encyclopedia of Machine Learning, Springer, 2010.

[7] Harris, Zellig, "Distributional Structure." Word 10 (23): 146–162, 1954.

[8] D. M. Randel, "The new Harvard dictionary of music." Cambridge, MA: Belknap Press, 1986.

[9] A. L. Uitdenbogerd and J. Zobel, "An Architecture for Effective Music Information Retrieval." In Journal of the American Society for Information Science and Technology, 2004

[10] J. Sivic and A. Zisserman, "Efficient visual search of videos cast as text retrieval." IEEE Transaction on Pattern Analysis and Machine Intelligence, Vol. 31, No. 4. 2009.

[11] E. Dupraz and G. Richard, "Robust frequency-based audio fingerprinting." In Proceedings of the IEEE International Conference on Acoustics, Speech and Siganl Processing, pp. 281-284, 2010.

[12] J. Musil, B. Elnusairi and D. Müllensiefen, "Perceptual Dimensions of Short Audio Clips and Corresponding Timbre Features." 9th International Symposium on Computer Music Modelling and Retrieval (CMMR), 2012.

[13] D. Tingle, Y. E. Kim and D. Turnbull, "Exploring Automatic Music Annotation with 'Acoustically-Objective' Tags." In Music Information Retrieval (MIR), 2010.

[14] T. Bertin-Mahieux, D. P.W. Ellis, B. Whitman, and P. Lamere, "The Million Song Dataset." In Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR), 2011.

[15] B. McFee, T. Bertin-Mahieux, D. P.W. Ellis and R.G. Lackriet, "The Million Song Dataset Challenge." International World Wide Web Conference Comittee, 2012

[16] T. Jehan and D. DesRoches, "Echonest Analyzer Document. v3.08." The Echo Nest Corporation, 2011.

[17] K. D. Martin, "Automatic Transcription of Simple Polyphonic Music: Robust Front End Processing." M.I.T. Media Laboratory Perceptual Computing Section Technical Report No. 399, 1996.

[18] A. Klapuri, T. Virtanen and T. Holm, "Robust multipitch estimation for the analysis and manipulation of polyphonic musical signals." In Proceedings of Conference on Digital Audio Effects, DAFx-00, Verona, Italy, 2000.

[19] D. Heckerman. "A Tutorial on Learning with Bayesian Networks." In Learning in Graphical Models, M. Jordan, ed. MIT Press, Cambridge, MA, 1999.

[20] X. Su and T. M. Khoshgoftaar "A Survey of Collaborative Filtering Techniques." In: Advances in Artificial Intelligence, 2009.

[21] P. Resnick, N. Iacovou,M. Suchak, P. Bergstrom, and J. Riedl. "GroupLens: An Open Architecture for Collaborative Filtering of Netnews." In CSCW '94: Conference on Computer Supported Coorperative Work, 1994.

[22] C. Basu, H. Hirsh, and W. Cohen, "Recommendation as Classification: Using Social and Content-based Information in Recommendation." In Recommender System Workshop '98. pp. 11-15, 1998.

[23] J.S. Breese, D. Heckerman D and C. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering." In: Proceedings of fourteenth Conference on Uncertainty in Artificial Intelligence. pp 43–52, 1998.

[24] D. Bogdanov, J. Serra, N. Wack, and P. Herrera, "From Low-level to High-level: Comparative Study of Music Similarity Measures." In International Workshop on Advances in Music Information Research, 2009.

[25] E. Pampalk, "Computational models of music similarity and their application in music information retrieval." PhD thesis, Vienna University of Technology, Mar. 2006.

[26] B. Logan, "Music recommendation from song sets." In Proceeding of Internet Society for Music Information Retrieval, page 425–428, 2004.

[27] B.Logan. and A. Salomon, "A Content-Based Music Similarity Function." Cambridge Research Laboratory Technical Report Series, Jun. 2001.

[28] E. Pampalk, "Speeding Up Music Similarity." In Proceedings of the MIREX Annual Music Information Retrieval eXchange, 2005.

[29] S. McAdams and A. Bregman, "Hearing musical streams." Computer Music Journal, 3(4), pp. 26-43, 1979.

[30] "American standard acoustical terminology (including mechanical shock and vibration)." American Standards Association, May 25, 1960.

[31] J. R. Hershey and P. A. Olsen, "Approximation of the Kullback Leibler divergence between Gaussian mixture models." International Conference International Conference on Acoustics, Speech and Siganl Processing, 1988.

[32] E. Gaussier, "Recommender systems in industrial contexts." phD thesis, University of Grenoble, 2012.

[33] D. Bogdanov and P. Herrera, "Taking advantage of editorial metadata to recommend Music." 9th International Symposium on Computer Music Modelling and Retrieval (CMMR), 2012.

[34] A. Nanopoulos, D. Rafailidis, P. Symeonidis, and Y. Manolopoulos, "MusicBox: Personalized Music Recommendation Based on Cubic Analysis of Social Tags." IEEE Transaction on Audio, Speech and Language Processing, Vol. 18, No. 2, Feb 2010.

[35] V. Zanardi and L. Capra. "Social Ranking: Finding Relevant Content in Web 2.0." International Workshop on Recommender Systems, 2008.

[36] C. D. Manning, H. Schütze. "Foundations of Statistical Natural Language Processing." Cambridge, MA: The MIT Press, 1999.

[37] M. Blum, R. W. Floyd and R. E. Tarjan. “Time bounds for selection.” Journal of Computer and System Sciences 7 (4): 448–461, 1973.

[38] M. Richardson, A. Prakash and E. Bill, “Beyond PageRank: Machine Learning for Static Ranking.” International World Wide Web Conference Committee, 2006.

[39] E. Bernardsson. “Music recommendation at Spotify.” Presentation at NYC Machine Learning, 2012

[40] “File:ADSR parameter.svg - Wikipedia, the free encyclopedia” Retrieved in 30.11.2013, from http://en.wikipedia.org/wiki/File:ADSR_parameter.svg

[41] “The Million Song Challenge | Million Song Dataset” Retrieved in 30.12.2013, from http://labrosa.ee.columbia.edu/millionsong/challenge

[42] “MIDI Tuning” Retrieved in 7.01.2013, from http://www.midi.org/techspecs/midituning.php

[43] S. Downie, “Music information retrieval.” Annual review of Information Science and Technology, 2003